CAPACITATED SCHEDULE-BASED TRANSIT ASSIGNMENT USING A CAPACITY PENALTY COST

by

HYUNSOO NOH

_____

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF CIVIL ENGINEERING AND ENGINEERING MECHANICS

In Partial Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

WITH A MAJOR IN CIVIL ENGINEERING

In the Graduate College

THE UNIVERSITY OF ARIZONA

2013

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Dissertation Committee, we certify that we have read the dissertation prepared by Hyunsoo Noh, titled Capacitated Schedule-Based Transit Assignment Using a Capacity Penalty Cost and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

_____ Date: January 6, 2014
Mark D. Hickman

_____ Date: January 6, 2014
Yi-Chang Chiu

_____ Date: January 6, 2014
Pitu Mirchandani

_____ Date: January 6, 2014
Kenneth L. Head

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

_____ Date: January 14, 2014
Dissertation Director: Mark D. Hickman

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of the requirements for an advanced degree at the University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that an accurate acknowledgement of the source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: Hyunsoo Noh

ACKNOWLEDGEMENTS

It was a long but jubilant journey to arrive at my first academic goal in Tucson. Above all, I would like to thank my advisor, Professor Mark Hickman, for his sincere support as a mentor, guidance as a scholar, and everlasting positive encouragements during my dissertation study.

I truly appreciate to my committee members, Professors Yi-Chang Chiu, Pitu Mirchandani, Larry Head, and Wei Hua Lin for their involvement and valuable comments. Regarding numerous opportunities at the University of Arizona and in Portland, OR, I thank Professor Yi-Chang Chiu again.

Many thanks to my transit team, Sang-gu Lee, Alireza Khani, Neema Nassir, and Andisheh Ranjbari for their proactive and positive involvement of the University of Arizona Transit Research Unit (UATRU). I also thank former and current transportation group colleagues, Hong Zheng, Brenda Bustillos, Eric Nava, Ye Tian, Xianbiao Hu, and Xueyan Du and Yiheng Feng. I would love to remember all the memories with them.

I would like to express my deepest appreciation to my wife, Yunemi, for her support and understanding in my pursuit of this long journey as a wife and colleague. All the achievement I have finished would not be possible without her compassionate care. Finally, I dedicate this work to Him, my friend and strength.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Schedule-based transit assignment models have been studied extensively from 2000, considering more time-dependent transit passenger behavior associated with the transit schedule. Currently, transit schedule information is more easily accessed using new telecommunications systems, such as mobile devices and the internet. One critical example of information sharing is Google's General Transit Feed Specification (GTFS). The information of the schedule *per se*, however, is not enough to explain the transit passenger's behavior, especially in a congested transit system. Regarding the congestion issues on a transit system, numerous researches have studied a transit schedule network (Nguyen et al., 2001; Nuzzolo et al., 2001; Poon et al., 2004; Hamdouch and Lawphonpanich, 2008, 2010).

Along the stream toward understanding transit passenger behavior in the capacitated transit schedule network, we propose solution models for solving the deterministic and stochastic user equilibrium (SUE) problems on a capacitated transit schedule network. Nguyen et al. (2001) introduced how the capacitated user equilibrium (UE) on a transit schedule network is different from the auto user equilibrium. For the foundation of the study, we utilize the link-based and time-expanded (LBTE) transit schedule network introduced by Noh et al. (2012a) which effectively captures turning movements like transfers easily as well as maintaining the efficient size of a schedule-based network. In the LBTE transit network, time points are assigned to each link connecting two stops by each run (or route). Utilizing the "link-based" structure, a link-based shortest path (LBSP) and hyperpath search (LBHP) models (Noh et al., 2012a) are introduced. Especially, the hyperpath employs a log-sum weighting function for incorporating multiple schedule alternatives at each stop node considering passenger's stochastic behavior.

One distinctive transit passenger behavior over a congested transit system is a first-in-first-out (FIFO) priority on boarding. A passenger already on board has the higher priority than passengers who are about

to boarding, and the passengers arriving earlier at a stop will have higher priority than the passengers arriving later at the stop. To consider the capacitated UE considering the relation between the FIFO boarding priority and vehicle capacity constraint, we apply a "soft-capacity" cost (Nguyen et al., 2001). This soft capacity cost function allows some violation of the predefined vehicle capacity, but the violation will be penalized and affect the cost of the path in the next iteration. The penalty of the soft capacity cost function allows not to assigning passengers on the alternatives having the lower priority of boarding, which finally leads to the solution of the capacitated transit deterministic user equilibrium (DUE) or SUE problems.

For the main transit assignment models, we proposed path- and hyperpath-based methods and a self-adaptive method considering deterministic and stochastic passenger behaviors. *First*, we developed the hyperpath-based assignment method by Noh et al. (2012b). For the FIFO transit passenger behavior, typically accompanying asymmetric (non-separable) cost relation, we also introduce a diagonalization technique (Sheffi, 1985) with the method of successive average (MSA) assignment technique. As expecting a better performance, *second*, we introduced the path-based assignment models using gradient projection. For the FIFO passenger behavior on boarding, we considered the same diagonalization approach used in the hyperpath-based assignment model and a full-Hessian scaling matrix in the gradient projection. By utilizing a full path set for each O-D pair, a better performance is guaranteed with the path-based model but the diagonalization technique may result in longer iterations. For improving the diagonalization steps, *third*, we explored several other possible methods. Above all, we proposed the better initial solution (BIS) model which assigns the initial flows on the priority path over congested links and also maintains feasible flows below the capacity constraint. On the other hand, we also added two additional assignment models to improve the diagonalization technique. One utilizes a full Hessian scaling matrix in the proposed path-based assignment model instead of diagonalization and the other is the self-adaptive gradient projection (SAGP) model introduced by Chen et al. (2012) which does not require a scaling matrix by optimizing the step-size in the path-based projection model. For improving the

SAGP model, we modified the SAGP model. First, we applied the SAGP at a disaggregate level for each O-D pair as expecting a compact set of path alternatives limited by each O-D pair, called disaggregate self-adaptive gradient projection (DSAGP). Second, we applied a type of diagonalization technique in the SAGP model by maintaining the residual capacities for the estimated flows in the next iteration.

Beyond just a single model development, the proposed transit assignment models not only showed various possibilities of the transit assignment, but also showed which model is more efficient and practical in terms of a real application. A computational model structure using the proposed models was mainly designed for an effective model development by sharing numerous components as well as maintaining the efficient data structure. The nine combination models based on the proposed three main models (hyperpath- and path-based and DSAGP assignment models) and the efficient BIS technique for solving the problems were tested and analyzed on a sample network and a partial Sacramento regional transit network.

1 INTRODUCTION

1.1 Background

For the past several decades, the frequency-based approach has dominated the development of transit assignment models. The frequency-based approach best captures passenger strategic behavior according to the random arrival pattern of a transit vehicle (Spiess and Florian, 1989) for high-frequency (or short headway) transit service in urban areas. Even though it maintains a route-level network resolution, the frequency-based approach allows a manageable memory size and complexity in its computational effort. Schedule-based transit assignment models have been studied extensively from 2000, considering more time-dependent transit passenger behavior associated with the transit schedule. Currently, transit schedule information is more easily accessed using new telecommunications systems, such as mobile devices and the internet. The evidence of increased access to transit information is closely related to the effort to share the public information. One critical example of information sharing is Google's General Transit Feed Specification (GTFS), gathering the transit schedule information from different service areas in U.S. This open environment for transit information makes passengers actively utilize the schedule information when making their transit trips.

The information of the schedule *per se*, however, is not enough to explain the transit passenger's behavior, especially in a congested transit system. There have been many success stories of public transit to support greater mobility and relieve roadway congestion, but this success also has created congested public transit systems, typically in high-density and developing urban areas. Regarding the congestion issues on a transit system, numerous researches have studied a transit schedule network (Nguyen et al., 2001; Nuzzolo et al., 2001; Poon et al., 2004; Hamdouch and Lawphonpanich, 2008, 2010). The proposed study

is along this stream, toward understanding transit passenger behavior in the capacitated transit schedule network.

1.2 Scope of the Study

Transit assignment is usually categorized to a frequency-based and a schedule-based model, according to the passenger's decision making in light of the transit schedule. This classification is shown in Figure 1.1. An aggregate service level will be considered for a "frequency-based" approach and disaggregate service level for "schedule-based" approach. In the frequency-based approach, the passenger can choose a route based on its headway, while the schedule-based approach allows the passenger to choose a specific vehicle trip. This study is interested in a schedule-based model. *First*, each transit vehicle run (trip from terminal to terminal) is the basic unit of resolution. The underlying assumption is that the passenger's route choice depends distinctively on each transit vehicle's schedule at a specific stop. *Second*, the vehicle capacity is a critical criterion for building the model, since the capacity may create a different passenger behavior based on the passengers' boarding priority. It is also assumed that the transit system has a certain level of congestion on board. When we consider this congestion, *third*, a capacitated transit assignment model can be categorized into two main models, namely hard- and soft-capacity. A hard-capacity model is defined by following the strict vehicle capacity, not allowing the flows to increase above a predefined capacity of each vehicle. On the other hand, a soft-capacity model allows some violation of the predefined vehicle capacity, but the violation will be penalized and will affect the cost of the path in the next iteration. For this study, we are interested in this soft-capacity form of a schedule-based transit assignment model.

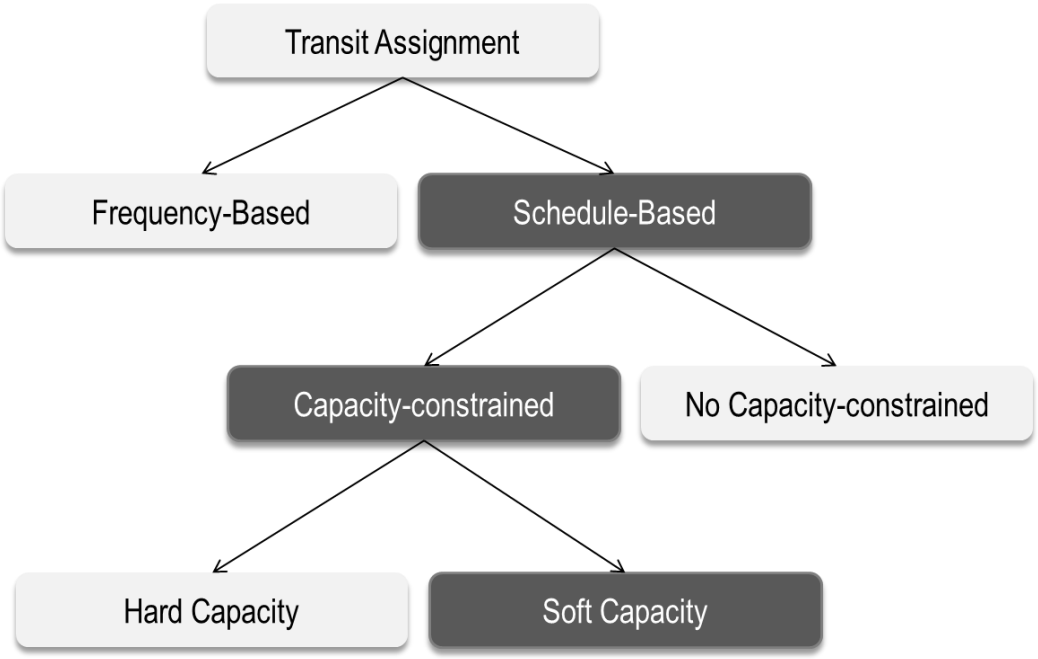


Figure 1.1 Scope of the Study

1.3 Problem Statement

1.3.1 An Example and Assumptions

With the schedule information from various sources, such as mobile device, internet, and printed schedule books, we may ask how a transit passenger makes his/her trip. For a more specific representation, let us assume a transit passenger is waiting for a transit vehicle at a stop, and the passenger would like to arrive within the preferred arrival time (PAT) at his/her destination. Of course, we assume that the passenger has the schedule information for his/her desired trip, and that the trip is being made in the AM peak period during transit congestion. Each passenger will follow his/her chosen discipline at each stop to arrive at his/her destination, and several alternative routes from the stop to the destination may be considered. When a specific transit run has arrived at the stop, he/she may then realize that there is insufficient

capacity to get on. The passenger denied boarding will decide to wait for the next transit vehicle or may choose any other available routes in order to arrive at destination satisfying his/her PAT. If the passenger arrived late at the destination, the passenger will probably choose an earlier alternative for the next day's trip in order to arrive at the destination in time. Finally, the chosen earlier alternatives will result in an earlier departure time.

If passengers know the schedule in order to make a trip, they will actively employ it until they find a better path, should they be denied boarding due to limited transit vehicle capacity. In this model approach, we will make some assumptions about the sample transit system. *First*, in terms of the schedule-based approach, it is assumed that passengers will respond to the fixed and static schedule. According to the schedule, each transit vehicle arrives and departs on time, and the number of passengers boarding and alighting does not affect the dwell time. *Second*, a passenger has either a preferred arrival time (PAT) at the destination or a preferred departure time (PDT) from the origin. *Third*, a passenger will minimize his/her total expected cost, which is represented in a generalized cost term considering transfer time, in-vehicle travel time, and walking time. *Fourth*, a group of passengers are assumed to have identical preferences in a deterministic model approach, but passengers are allowed to have different perceptions of the expected travel cost in a stochastic model approach. *Fifth*, in the model, capacity violations are allowed. We call this relaxation a "soft" capacity model, typically using a monotonically increasing penalty function that varies with the demand and with the available residual capacity. Regarding these assumptions, we will explore the relationship between a passenger's priority and the user equilibrium (UE) problem on a transit schedule network.

1.3.2 Passenger Priority

Each passenger will have his/her priority for boarding according to his/her arrival time at a stop or at the given boarding location. Nguyen et al. (2001), Hamdouch and Lawphonpanich (2008, 2010) and Poon et al. (2004) mention this transit FIFO rule: on a transit schedule network, a passenger already on board has the higher priority than boarding passengers, and the passengers arriving earlier at a stop will have higher priority than the passengers arriving later at the stop. This priority behavior rule can be represented in a transit schedule network as shown in Figure 1.2.

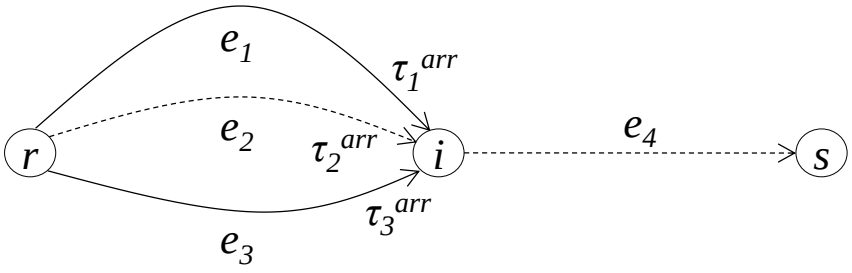


Figure 1.2 Passenger Priority Representation

In Figure 1.2, we assume that three links (e_1, e_2, and e_3), with arrival times ($\tau_1^{arr} < \tau_2^{arr} < \tau_3^{arr}$) at node *i*, are connected to link e_4 and the transit vehicle runs of e_2 connect directly to e_4 (the dashed lines), which includes transfer times $t_{e2e4}^{trsf} = 0$, $t_{e1e4}^{trsf} > 0$, and $t_{e3e4}^{trsf} > 0$. The passengers on link e_2 will have first priority to proceed to link e_4 since they are already on board. The second and third priority is ordered by the arrival time, $\tau_a^{arr} + t_{ab}^{trsf}$ of each vehicle from link e_1 and e_3. This order of priority will then be used to manage the boarding process when there is a limited transit vehicle capacity.

1.3.3 Capacitated User Equilibrium (UE) on a Transit Schedule Network

The capacity of each vehicle and the priority of passengers generate a different passenger behavior, especially in a congested network like that shown in Figure 1.2. We can think about a user equilibrium with this specific passenger behavior based on a capacitated UE problem given by Nguyen et al. (2001). Assume that the network has three different routes, that 21 passengers will be assigned to the origin-destination (O-D) pair, and that each link has its given cost and capacity. The anticipated system optimal (SO) result is depicted in Figure 1.3(a). Every passenger will use the upper-most route, up to its capacity, and a single passenger will choose the bottom route to minimize the total network cost. However, if this single passenger wants to reduce his/her disadvantage, the passenger will use the second (middle) route, since this has shorter time and has priority over the top-most route. This would displace one other passenger from the top-most route, who again may shift to the middle route (and then would displace another passenger). Finally, all passengers will then shift to the second route until no more capacity is available. The main reason for moving from the solution in (a) to the solution in (b) is the path priority. The second route dominates the first (upper) route, because it has priority to proceed onto link (*i*,D).

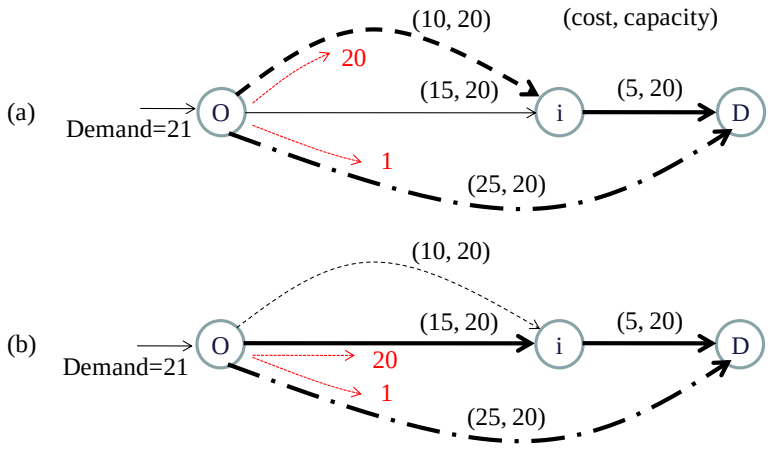


Figure 1.3 (a) SO and (b) UE on a Capacitated Transit Network

For this problem, Nguyen et al. (2001) considered the priority of the route to apply a penalty cost for a transfer from one route to another, in order to generate a deterministic user equilibrium (UE). The penalty cost function in that study was a type of power function, although they also mentioned the difficulty of practical application with their functional form of the penalty cost.

1.4 Objective of the Study

The main objective of this study is to solve the predefined problem considering transit passenger behavior on a transit schedule network and a soft capacity constraint on the transit vehicle. To accomplish this objective, several critical questions are:

- How should we represent the transit schedule network?
- How should we search the optimal paths for transit passengers on the proposed transit network?
- How should we assign transit passengers with the proposed path models on the transit schedule network, especially with the vehicle capacity constraint and the priority of boarding?
- How can both deterministic and stochastic assignment models be applied for this passenger assignment?

Regarding the objective and associated questions of the study, certain intermediate objectives can be established:

- *First of all*, a transit schedule network is an important foundation for searching paths and assigning transit passenger on the searched paths. This intermediate objective focuses on an easier representation of the transit schedule network that can still capture transit passenger behaviors as well as suitable path models.

- The *second* intermediate objective is to provide optimal path models. The considered path models are a single shortest path model and a hyperpath model. The expectations on the proposed path models are: (1) compatibility with the proposed transit schedule network, satisfying the time-dependent constraint using schedules along consecutive stops; and, (2) efficiency in its computational performance.
- *Third*, this study will focus on the proper passenger behavior under the relation between the vehicle capacity constraint and passenger's priority. To link the relation to the assignment models, a soft capacity assignment technique is considered to penalize the lower priority flows on the capacitated network.
- In the transit assignment, *fourth*, we explore two possible assignment models utilizing the proposed path models. One uses a schedule-based hyperpath model and the other uses existing auto assignment models, but expanding to stochastic assignment models. The considered model is based on the so-called *path-based* assignment approach.

1.5 Importance and Challenges of the Study

First, because it is based on utilizing transit schedules, each passenger's behavior can be traced to an individual transit vehicle. This representation allows more realistic model representation of passengers' decision making, with specific space and time resolution. By tracing the passenger movements associated with transit vehicle movements, it is possible to model a time-dependent boarding and alighting behavior with a specific vehicle, accounting for explicit vehicle capacity.

Second, transit assignment with a large transit data set, stemming from a large urban transit network, can be a challenge in its application, because the time-expanded transit schedule network can grow quickly in terms of temporal and spatial expansion just by adding several routes. Also, it is rare to find similar large-scale applications for this same reason. To resolve these challenges, we may need to consider existing

efficient models for both auto and transit assignment. If we can borrow efficient auto assignment techniques and apply them to transit assignment, it will create extensions such as intermodal assignment. However, for transit, we need to be careful to incorporate the distinctively different transit passenger behavior with respect to the vehicle capacity and priority. Although the passenger behavior and transit network are different from the auto driver's behavior and the road network, it is meaningful to investigate the possibility of using the existing auto assignment algorithms on a transit schedule network in order to generate reasonable computation performance.

Third, the time-dependent transit passenger assignments are critical to incorporate with other time-dependent transportation models, such as activity-based, agent-based, and dynamic traffic assignment (DTA) models. By explaining detailed time-dependent travel behavior, a transit assignment model on a transit schedule network connects to many other transportation models developed in a time-dependent environment. Since every decision is made at the level of each individual transit vehicle and at the level of the individual traveler, exact arrival and departure time of transit vehicles at a stop is estimated, along with the departure and arrival times for the passengers. Also, when we consider an intermodal assignment model, such as park-and-ride, kiss-and-ride, and bike-and-ride, to represent the intermodal assignment, the passenger's departure and arrival times are anchor points connecting to other transportation modes. Moreover, considering various routes as different modes, the transit assignment model can be easily applied to a multimodal network.

Fourth, this capacitated, schedule-based model is also critical to understand the detailed passenger behavior. With the development of transit information systems, especially for real-time applications, the time-dependent passenger behavior can be understood at a more detailed level, explaining transit vehicle choice as it relates to the realized schedule information.

1.6 Organization of the Study

To embody the objectives and importances, this study organizes chapters as shown in Figure 1.4.

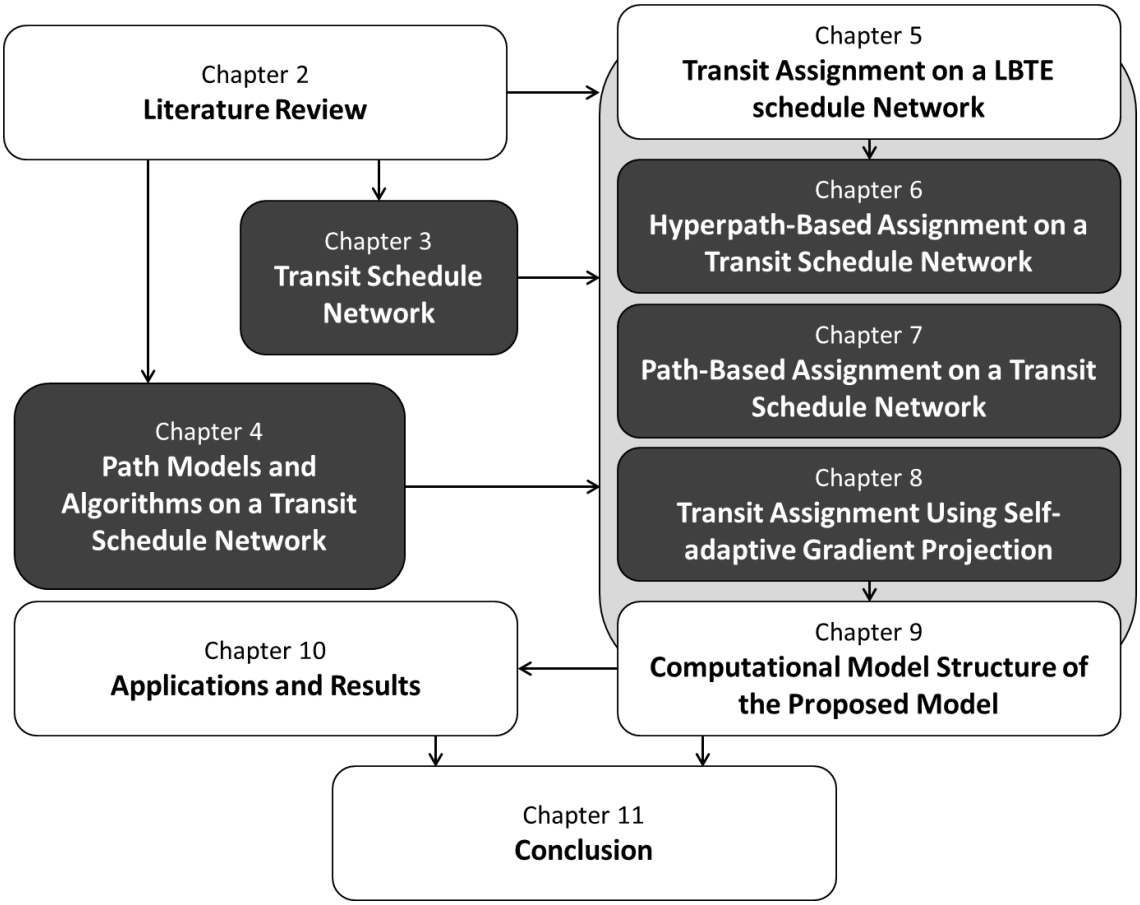


Figure 1.4 Organization of the Study

In Chapter 2, *first*, we explore the existing studies associated with the predefined objectives. According to the literature review, *second*, this study consists of critical model development cornerstones regarding how to prepare a transit schedule network (Chapter 3), how to develop the efficient path search models (Chapter 4) and major assignment model development (Chapter 6 ~8) including basic behavioral assumptions, capacitated user equilibrium, and initial feasible solution (Chapter 5). In Chapter 3, we introduce a link-based and time-expanded (LBTE) transit schedule network. With respect to the LBTE

transit network, we propose efficient shortest path and hyperpath models in Chapter 4. With the LBTE network and the proposed path search methods, we introduce three major transit assignment models considering deterministic and stochastic behavior: hyperpath- and path-based assignments (Chapter 6 and 7) and the transit assignment using self-adaptive grardient projection (Chapter 8). *Third*, the proposed assignment methodologies are embodied in the proposed computational model structure in Chapter 9. The overall computational model structure, detailed parameters, input and output formats are included in this chapter. Using the proposed computational model, *fourth*, we test and analyze the transit assignment models on a simple network and a partial transit schedule network of Sacramento resgion in Chapter 10 and we conclude this study as well as future works in Chapter 11.

In solving the transit assignment problem, as discussed in Chapters 6, 7, and 8, the objective of the study is mainly on solving the deterministic and stochastic transit assignment problems on a transit schedule network. In terms of solving the two problems, we have developed several heuristic solution methods with different performance characteristics, especially for the computation performance in a real application. *First*, we develop the hyperpath-based assignment method proposed by Noh et al. (2012b) in Chapter 6. For the FIFO transit passenger behavior, typically accompanying asymmetric (non-separable) cost relations, we also introduce a diagonalization technique. From several initial tests, the hyperpath model had slower performance than other shortest path models, and the proposed MSA technique was also slower than other assignment techniques like Newton-type optimization techniques. For these reasons, *second*, we introduce the path-based assignment models using gradient projection in Chapter 7. For the FIFO passenger behavior on boarding, we consider the same diagonalization approach. By utilizing a full path set for each O-D pair, we realized that better performance is guaranteed with the path-based model, but the diagonalization technique may result in longer iterations. To improve this inefficient diagonalization, *third*, we explored several other possible methods. Above all, we proposed the better initial solution (BIS) model, which assigns the initial flows on the priority path over congested links and also maintains feasible flows below the capacity constraint. On the other hand, we also tested two

additional assignment models to improve the diagonalization technique. One utilizes a full Hessian scaling matrix in the proposed path-based assignment model instead of the diagonalization in Chapter 7, and the other is the self-adaptive projection method, in Chapter 8, which does not require a scaling matrix by optimizing the step-size in the path-based projection model.

2 LITERATURE REVIEW

Considering the previous objectives, we explore the existing research in terms of:

- the network representation upon which to build the transit path and assignment models,
- the path search models typically utilized in the transit assignment for both frequency- and schedule-based approaches, such as the hyperpath and the shortest path models,
- the frequency- and schedule-based transit assignment models for vehicle capacity constraint, and passenger deterministic or stochastic behavior, including path search models, and
- the additional auto assignment and route choice models considering stochastic passenger behavior.

2.1 Transit Network Representation

A transit network can be represented by various forms, depending on the needs of the models of passenger behavior and transit vehicle movements. Transit network assignment is categorized by two approaches: schedule- or frequency-based approach. The former uses the disaggregate schedule network representation that each schedule time is represented by a node or a time point at a physical stop. We explore this representation of the transit schedule network.

On the transit schedule network representation, most studies have represented a schedule at a transit stop by expanding each physical stop for each point in time as shown in Figure 2.1, in which stops are expanded by multiple “boarding/alighting” points. Tong and Richardson (1984) proposed a transit network structure which contains stops (or nodes), and every link connects two consecutive nodes, which are categorized into transit links and walk links. The network representation is the same as that in the frequency-based approach, except that a series of transit arrival times is given at each stop is typically formed by expanding nodes by the schedule time. Carraresi et al. (1996) introduced a prototype of a

space-time network for passenger assignment on a transit schedule service. Tong et al. (1999) also applied this node expansion by time in their proposed model. Considering the spatial and temporal range from origin to destination, Nuzzolo (2001) introduced a “diachronic” network (see Figure 2.1) which represents more detailed categorization of link and node sets as: run/dwell links, boarding and alighting links, access and egress links, and transfer links for the link set; and centroids, origin and destination time expansion nodes, stop nodes, boarding and alighting nodes, and vehicle run nodes for the node set. Instead, Poon et al. (2004) applied a hierarchical node structure. Each station node includes a platform node and a transit node, and the link set is divided into three types: a transit link between two transit nodes, a boarding link between the platform and the transit node, and walking links including access, egress, and transfer.

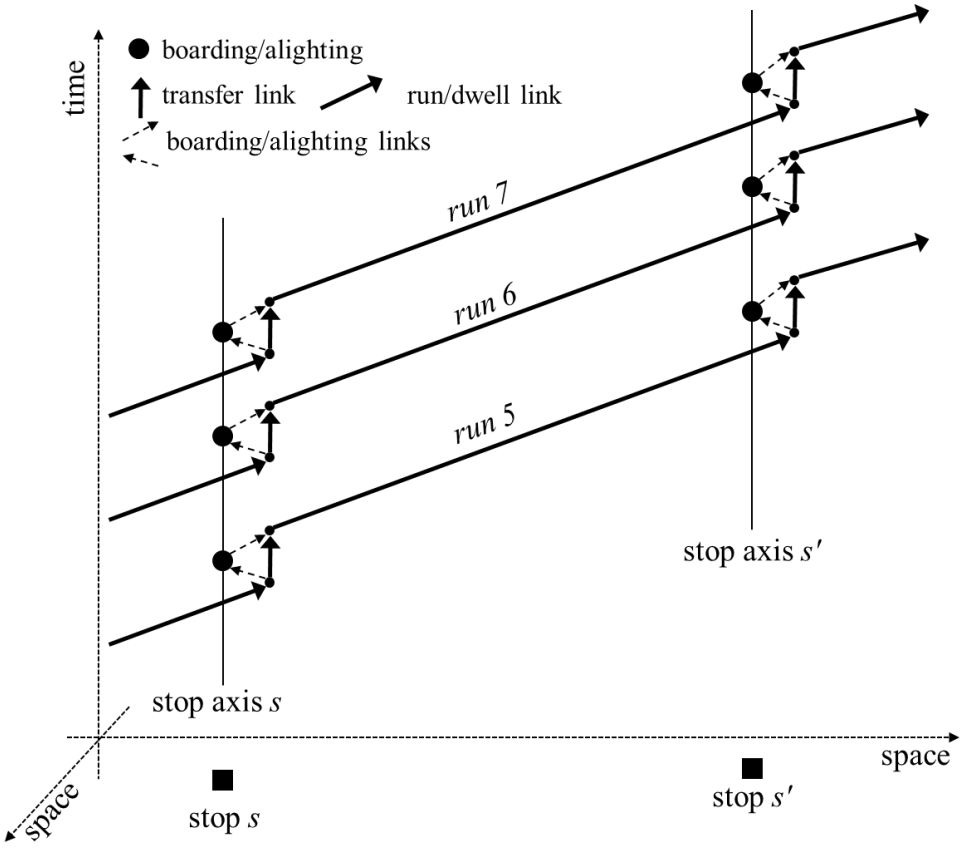


Figure 2.1 Stop Expansion according to Schedule Time (Nuzzolo, 2001)

To apply the strategy in a transit schedule, Hamdouch and Lawphonpanich (2008) proposed a time-expanded network in which each node is a time-expanded physical stop, an origin, or a destination. To achieve a strategy at a stop (a node), they introduced a waiting link, connecting two consecutive temporal nodes, as well as walking and vehicle run links. Nielsen and Frederiksen (2009) also represented a transit schedule network in terms of temporal node expansion. On the other hand, Noh et al. (2012a) introduced a different type of time expansion, called a "link-based and time-expanded (LBTE)" transit network as shown in Figure 2.2. Instead of expanding a node by the schedule time, each route segment between two consecutive stops is expanded to create multiple links, each link representing each run of the route between the two stops. Most existing transit schedule network representations are closely linked to a least-cost path model that updates the label of the cost on each node. Instead, Noh et al. (2012a) mainly utilized link-labels by updating each link label while searching a path.

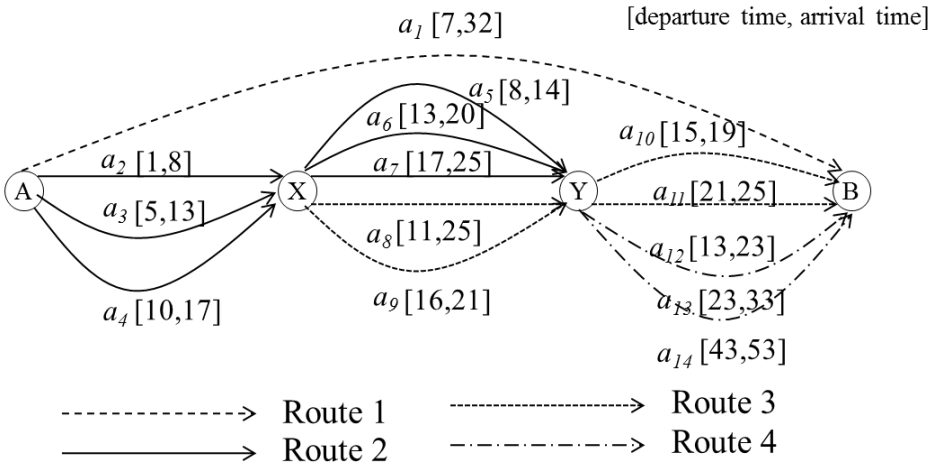


Figure 2.2 Link-Based and Time-Expanded (LBTE) Transit Network (Noh et al. 2012a)

2.2 Path Search Models

With this transit network representation, we explore the existing path models. Paths on a transit network can be searched simply by label-setting or label-correcting shortest path models or a shortest hyperpath model, and these can be applied to a time-expanded transit schedule network or on a frequency-based transit network. *First*, on the frequency-based approach, the shortest path model was applied by Dial et al. (1967) on a trunk-link line (TLL) transit network, using Moore's algorithm (a label-correcting method). Le Clercq (1972) introduced an improved shortest path model based on Dial et al. (1967), and De Cea and Fernández (1993) utilized a shortest path algorithm in a Frank-Wolfe assignment model. On a transit schedule network, Tong and Richardson (1984) proposed a generic one-to-one *optimal* shortest path algorithm with a generalized cost form, which defined a schedule-based "efficient" path based on Dial's (1971) definition of an efficient path. Tong and Wong (1999), Tong et al. (2001), and Poon et al. (2004) also employed this optimal path algorithm on a transit schedule network. Pallottino and Scutellá (1998) proposed Chrono-SPT which was developed as a dynamic path algorithm using a time-space expanded network; this algorithm satisfies the optimality conditions under a time-connection constraint. Nguyen et al. (2001) also utilized the Chrono-SPT model to search for a path on a transit network. Nielsen and Frederiksen (2006, 2009) introduced several algorithms for a transit schedule network: label-correcting, label-setting, rule-based, and a mixed version of the algorithm. Noh and Hickman (2012) applied a similar type of Chrono-SPT model, but using a link-based labeling approach.

Second, the idea of hyperpath was originally defined and introduced by Nguyen and Pallottino (1988, 1989) and Spiess and Florian (1989), where the latter applied the concept to the frequency-based transit assignment considering a passenger's strategy, with randomly arriving transit vehicles. The final outcome is a hyperpath consisting of a set of alternative paths from origin to destination. You can see other frequency-based hyperpath models in Table 2.1. Gallo et al. (1993) generalized the hyperpath model as defining a fundamental network structure, not just including the "strategy" concept used in the

frequency-based approach. Beyond a simple hyperpath model, Nielsen (2001) and Nielsen et al. (2005) introduced a *K*-hyperpath model which follows the basic hyperpath definition of Gallo et al. (1993).

The hyperpath model was also applied to a transit schedule network in some recent works. Each path model is closely associated with a specific network representation. Hamdouch and Lawphonpanich (2008, 2010) employed a hyperpath model for passenger loading. Rochau et al. (2010) introduced a hyperpath methodology as searching nodes in temporal schedule order. As commented earlier, most of studies utilize the node labeling on a node-expanded transit schedule network. The link-based hyperpath model was introduced by Noh et al. (2012a) which followed a similar temporal schedule approach of Rochau et al. (2010).

Table 2.1 Existing Path Models on a Transit Network

	Shortest path	Strategy or Hyperpath
Frequency-Based Transit System	Dial et al. (1967) Le Clercq (1972) De Cea and Fernández (1993)	Chriqui and Robillard (1975) Nguyen and Pallottino (1988, 1989) Spiess and Florian (1989) Wu et al. (1994) Cominetti and Correa (2001) Cepeda et al. (2006)
Schedule-Based Transit System	Tong and Richardson (1984) Tong and Wong (1999) Tong et al. (2001) Poon et al. (2004) Nguyen et al. (2001) Nielsen and Frederiksen (2006, 2009) Noh and Hickman (2012)	Hamdouch and Lawphonpanich (2008, 2010) Rochau et al. (2010) Noh et al. (2012a)

2.3 Transit Assignment Models

2.3.1 Frequency-Based Transit Assignment

To consider the studies of schedule-based transit assignment models, it is worth mentioning the frequency-based transit assignment studies in terms of a critical foundation on transit passenger behavior. The early path models were studied by Dial (1967) and Le Clercq (1972). A passenger's *strategic* behavior in a "common line" problem, representing multiple transit routes as one simple route with higher frequency or lower waiting time, was introduced by Chriqui and Robillard (1975). On this fundamental foundation of passenger behavior, Nguyen and Pallottino (1988) and Spiess and Florian (1989) introduced important transit assignment models emphasizing the passenger's strategic behavior as a "hyperpath" or "strategic path". In these transit assignment models, transit vehicles are assumed to arrive randomly and each passenger chooses the first arriving vehicle in his/her preference set of routes. Nguyen and Pallottino (1988) set up a user equilibrium (UE) model in which the hyperpath flows satisfying a certain variational inequality (VI) reach a user equilibrium (all passengers experience their minimum cost). In the same context, Spiess and Florian (1989) introduced an assignment model minimizing the total expected travel times, creating a so-called "optimal strategy" between every origin-destination pair. At every stop node, the minimum (or optimal) expected travel time is estimated for all possible routes to the destination, and flows at each stop are assigned by the frequency ratio. The algorithm traces the hyperpath by a backward searching algorithm, a Dijkstra-type optimal path. A more detailed taxonomy of frequency-based assignment models is shown in Table 2.2, with further subdivision based on considerations of congestion, capacity, and the use of a single path or hyperpath.

Table 2.2 Frequency-based Transit Assignment Models for Passenger Behavior and Capacity

Frequency-based Model		Shortest path	Strategy or Hyperpath
Passenger Behavior on route choice	Deterministic	De Cea and Fernández (1993)	Nguyen and Pallottino (1988) Spiess and Florian (1989) Wu et al. (1994) Cominetti and Correa (2001) Cepeda et al. (2006)
	Stochastic	Nielsen (2000) Lam et al. (1999, 2002)	
Congestion on passenger boarding	Hard capacity	Lam et al. (1999, 2002)	
	Soft capacity		De Cea and Fernández (1993) Wu et al. (1994) Cominetti and Correa (2001) Cepeda et al. (2006)

Beyond the deterministic and non-capacitated frequency-based assignment models that typically utilize "strategic path" or "hyperpath" for the passenger behavior, the next issue was a capacity consideration, using "hard" and "soft" capacity. Hard capacity does not allow to having a violation; the soft capacity model allows that passengers exceed the given vehicle capacity, but with high "congestion" costs. De Cea and Fernández (1993) introduced a UE model incorporating a congestion effect, using an "effective frequency" using a flow-dependent waiting time. To solve the (soft) capacitated assignment problem, De Cea and Fernández (1993) proposed a type of Frank-Wolfe algorithm using diagonalization. Wu et al. (1994) proposed a symmetric linearization method using the linearized Jacobi method with a hyperpath model to solve the soft capacity UE problem. Both studies by De Cea and Fernández (1993) and Wu et al. (1994) showed a "semi-asymmetric flow dependency" where the waiting time increases by flows on other links. However, in this solution method, the flows are split only by the frequency ratio. On the other hand, Cominetti and Correa (2001) introduced "semi-priority-dependent" equilibrium model using the flow-dependent frequency model. In continuing the work of Cominetti and Correa (2001), Cepeda et al. (2006)

proposed a large-scale equilibrium model using the method of successive averages (MSA) technique. Additionally, considering a passenger's stochastic behavior on a frequency-based transit system, Nielsen (2000) and Lam et al. (1999, 2002) introduced stochastic assignment models, probit- and logit-type respectively.

2.3.2 Schedule-Based Transit Assignment

Beyond the basic optimal path model on a transit schedule network by Tong and Richardson (1984), schedule-based assignment models were initiated in the middle of the 1990s (see Table 2.3). To consider feasible passenger flows, Carraresi et al. (1996) clarified several meaningful definitions, such as feasible paths under a vehicle capacity constraint; a passenger's preferred departure and arrival time for a transit trip; and, the generalized cost and disutility of travel time. Based on the optimal shortest path model of Tong and Richardson (1984), Tong and Wong (1999) introduced a transit assignment model with the detailed time-dependent loading, while relaxing the capacity constraint.

Table 2.3 Schedule-based Transit Assignment Models for Passenger Behavior and Capacity

Schedule-based Model		Shortest path	Strategy or Hyperpath
Passenger Behavior on route choice	Deterministic	Carraresi et al. (1996) Tong and Wong (1999) Nguyen et al. (2001) Poon et al. (2004) Noh and Hickman (2012)	Hamdouch et al.(2004, 2008, 2010)
	Stochastic	Nuzzolo et al. (2001) Nielsen and Frederiksen (2006) Noh and Hickman (2012)	Noh et al. (2012b)
Congestion on passenger boarding	Hard capacity	Carraresi et al. (1996) Poon et al. (2004)	Hamdouch et al.(2004, 2008, 2010)
	Soft capacity	Nguyen et al. (2001) Nuzzolo et al. (2001) Noh and Hickman (2012)	Noh et al. (2012b)

Schedule-based transit assignment was more widely studied in the 2000s. Capacity in the schedule-based transit assignment is defined by a direct vehicle capacity (this differs from the flow-based capacity in a frequency-based transit system. The definitions of hard and soft capacity in this case are strictly dependent on the treatment of each vehicle's possible capacity violation. Of course, the hard capacity model does not allow violation of a defined vehicle capacity. Nguyen et al. (2001) fundamentally introduced a new UE definition regarding the priority of passengers under a capacitated transit schedule network. To consider this priority effect, they assume that the passengers' departure time and preferred arrival interval are given, and "soft" capacity constraints are applied. For the solution algorithm, disaggregate simplicial decomposition (DSD) was utilized (Larsson and Patriksson 1992). Nuzzolo et al. (2001) introduced a doubly dynamic assignment considering both "within-day" and "day-to-day" assignment procedures together, especially considering "with information" and "without information" cases, and considering a choice model both for stop and for route. The study assumed that demand between each origin-destination (O-D) pair is fixed at each point in time and is independent in the day-to-day evolution. A passenger has an attractive set of runs at each stop, which is dependent on his/her arrival time at the stop, and any run in the attractive set is connected to the destination. A soft capacity penalty is applied as a "discomfort" function within the path utility. Flows are assigned based on the utility of each path at the specific time. For day-to-day dynamic loading, several methodologies are recommended: fixed-point theory for regular service (deterministic), a Markov process for irregular service (stochastic), and Monte Carlo simulation for dynamic simulation.

Poon et al. (2004) introduced a dynamic and schedule-based assignment model considering a capacitated transit network. The first-in, first-out (FIFO) rule is applied under congestion, creating asymmetric costs of boarding and alighting arcs, where the costs of boarding and alighting depend on earlier boarding flows on previous moving links and on earlier waiting flows on previous boarding links. Time-dependent demand is given for each time interval, and vehicle capacity is a hard constraint. The day-to-day process is updated by previous experienced travel times and costs. A specific UE condition is defined; however,

since the constructed objective function is non-convex and not differentiable, the method of successive averages (MSA) is proposed to solve the problem. The algorithm used for searching the (optimal) weighted shortest path is based on Tong and Richardson (1984). In network loading, residual paths are used for the passengers on a boarding link who are denied boarding by the hard capacity constraint. This approach may be seen in contrast to the study by Nguyen et al. (2001) in terms of the capacitated UE, since a residual path will keep generating a feasible solution that does not violate the capacity constraint. As a result, in Poon's method, passengers will not be induced to change their routes to get higher priority. The loading process is divided among walking, boarding, and transit links. For termination, the (relative) gap function introduced by Smith (1993) is used, which is specified as the difference between the cost for the optimal path and the cost of other alternative paths generated between an O-D pair.

Different from other researches, Hamdouch et al. (2004, 2008, 2010) tried to understand the meaning of strategy in a transit schedule network according to the fundamental behavioral assumption of a strategy on a capacitated network (Marcotte et al., 2004). Passenger boarding follows a FIFO discipline or a random rule and each passenger or a group of passengers has a strategy that consists of an ordered preference set at each time-space node. At the starting nodes, passengers can be grouped by each strategy. The vehicle has a hard capacity constraint, such that passengers can be denied boarding because of the on-going priority and previous waiting passengers on the boarding link. Based on the passenger's strategy, a preference set at each node is defined, but allowing multiple destination time-space nodes. The expected strategy cost is generated by summing up each elementary path probability multiplied by the generalized cost of the elementary path in the strategy. To solve for the UE, MSA is applied and a strategy is used according to the available residual capacity and the priority of boarding. The optimal strategy is searched for all origin-destination (O-D) pairs with a typical gap function in MSA. For the optimal strategy, a backward search technique is maintained along the optimal preference set at each node, and the passenger loading process strictly follows each vehicle capacity.

Noh et al. (2012b) introduces a MSA-based assignment using a link-based hyperpath, incorporating a logit-type route choice behavior model, instead of considering a strategy on a transit schedule network like Hamdouch et al. (2004, 2008, 2010). To consider the asymmetric link cost in passenger boarding, Noh et al. (2012b) applied a diagonalization technique. In the congestion constraint, a soft capacity function is introduced similar to Nguyen et al. (2001) considering the relation among assigned flows, residual capacity, and the priority of boarding.

2.3.3 Auto Assignment Models

For transit assignment, we may consider using the existing auto assignment methods, especially those for static auto assignment and those that use a BPR-type penalty function for the capacity constraint. The representative auto assignment model considered in this study is a path-based model. Instead of using a classical link-based assignment algorithm like the Frank-Wolfe algorithm, Jayakrishnan et al. (1994) proposed a path-based assignment model which restates Beckman's UE objective and constraints in terms of non-negative, non-shortest paths, according to the Goldstein-Levitin-Poljak gradient projection by Bertsekas (1976). This takes the form of a quasi-Newton approximation. Sun et al. (1996) introduced a variant of the gradient projection model for a large automobile network. Chen and Lee (1999) and Chen et al. (2002) applied the gradient projection model and compared it to another path-based assignment model, disaggregate simplicial decomposition (DSD) introduced by Larsson and Patriksson (1992). Another possible model for transit assignment utilizes the Lipschitz condition for mildly asymmetric costs. He et al. (2002) introduced a self-adjusting step-size model to capture asymmetric costs, based on the work of Bertsekas (1976) which utilizes the projection model under the Lipschitz continuous conditions with Armijo (1966)'s self-adjusting step-size technique. Following these existing works, Zhou and Chen (2003) introduced a relaxed model compared to He et al. (2002) by embedding the Lipschitz constant and a strong monotone modulus in a modified form. On this fundamental foundation, Chen et al. (2012) introduced a self-adjusting step-size model especially for the non-additive cost function; this approach

incorporatesing the advantage of Bertsekas' Armijo rule with path-based gradient projection based on non-shortest path flows, which is now called the "self-adaptive" gradient projection (SAGP).

2.3.4 Traveler Behavior in a Stochastic Assignment model

For travel behavior, Dial (1971) introduced a stochastic assignment model based on the logit discrete choice model, called the STOCH algorithm, considering a random utility model on passenger's choice. To overcome the primary shortcoming of a logit-type model, also known as the Independence of Irrelevant Alternatives (IIA) property, Daganzo and Sheffi (1977) proposed a probit-based stochastic user equilibrium (SUE) model considering a non-separable cost function. However, the computational challenges of the probit model (requiring numerical integration) have resulted in the logit-based discrete choice model being more popular. Using the generalized extreme value (GEV) class, Vovsha (1997) developed a cross-nested mode choice model in a multimodal transportation network. The proposed model included a calibration procedure and application. Based on the random utility theory, Ben-Akiva and Bierlaire (1999) introduced a family of logit-type route choice models considering the traveler's perception of route overlapping. They also proposed hybrid logit models using a factor-analytic approach. Prashker and Bekhor (1998, 2000) explored SUE assignment models. The route choice models considered in Prashker and Bekhor (1998) were the C-logit, cross-nested logit (CNL), and paired combinatorial logit (PCL) models, and these were tested and compared on several test networks. The cross-nested model was chosen as the best stochastic loading model in the study since it does not require path enumeration.

Hoogendoorn-Lanser et al. (2005, 2007) focused on the multimodal route choice using the path-size logit correction for IIA. Hoogendoorn-Lanser et al. (2005) showed that the number of trip legs, instead of time or distance, is substantially better to represent the passengers' route choice with path overlaps for inter-urban trips with a main transit mode. These results were compared to the overlapping units of time and distance in various alternative models with different path size parameters. Hoogendoorn-Lanser et al. (2007) conducted an in-depth study of the path-size route choice model comparing several *separate path*

size logit (separate PSL) alternatives. Continuing the previous research from 2005, they applied three types of trip legs (access, egress, and main trip legs) on an inter-urban multimodal trip. Through the study, they showed that the separate PSL outperforms other methods in explaining the passengers' route choice behavior. Prato (2009) reviewed path choice set generation models using the shortest or k-shortest path algorithm and other stochastic path generation models.His study defined and compared (1) C-Logit, path-size logit (PSL), and path-size correction logit (PSCL) based on the logit foundation, and (2) PCL, CNL, and generalized nested logit (GNL) models for GEV foundation; and, (3) multinomial probit, logit kernel (LK) with random coefficients or with factor analytic approach on a non-GEV type foundation. Prato (2009) concluded that each model has advantages and disadvantages in terms of its theoretical background and application. The models categorized in (1) such as C-Logit and PSL have the advantages in their simple forms and relatively easy parameter estimations to apply on a large network but theoretically inferior to the models in categorized in (2). On the other hand, the models in (2) are challenging in terms of estimating the coefficients. However, the models in (2) are better in their performances than the non-GEV type models which require an additional simulation task although the non-GEV type models provide sound theoretical background.

2.3.5 Stochastic User Equilibrium Models

Based on the trip distribution model introduced by Evans (1973), Fisk (1980) constructed a logit-based SUE formulation by adding a path-based entropy term in the objective function, and she showed that the resulting objective function produces a solution that follows the logit model. This path-based entropy term was also used by Bekhor and Toledo (2005) in defining a SUE model with the path-based assignment approach. To overcome the IIA assumption of the logit model, Prashker and Bekhor (2000) proposed specific SUE models based on the Fisk's SUE model: basic multinomial logit, cross-nested logit, and paired combinatorial logit model. Xu et al. (2012) introduced path-based algorithms to solve SUE

problem, using a self-adaptive gradient projection method of Chen et al. (2012) with the C-logit model to correct for the path overlapping problem.

2.4 Anticipated Contributions of the Study

According to this review of the literature, we consider the following main problems for the contributions of this dissertation.

- *Time-expanded transit schedule network representation*

  A time-expanded network for a schedule-based transit assignment is a critical element to describe transit passenger behavior. Generally, the time-expanded network is represented in terms of "node expansion" in most studies. In the representation of the transit network, this study applies a "link expansion" based on Noh et al. (2012a) which creates a simpler transit schedule network and also accommodates the passenger behavior with a vehicle capacity constraint.

- *Passenger's priority behavior considering vehicle capacity and path search models*

  If a transit network is not congested, the problem of transit assignment should be as simple as sending passengers along their shortest paths or hyperpaths, or along a defined "strategy". But, if the vehicle capacity is limited and enough demand is loaded on the transit network, the problem will be challenging since passengers encountering vehicles at capacity will react differently than passengers in the non-capacitated case, especially on a relatively large transit schedule network. Regarding this problem, we consider "soft capacity" constraints and existing path search models like Dijkstra-type shortest path or hyperpath models on a transit schedule network. We also utilize a generalized cost on the searched paths.

- *Other possibilities to use the existing models used in auto and transit assignment*

  When we use a capacity penalty function, it is possible to consider other solution methods typically used in transit and auto assignment. One method extends the typical hyperpath solution method to schedule-based transit assignment. Another approach uses existing auto assignment methods. For this study, a path-based model and a self-adaptive gradient projection model based on the Lipschitz continuous condition are considered, explicitly including the vehicle capacity and the priority of boarding.

- *Large network application*

  In the literature review, large network applications are limited, especially using the transit schedule network, since the number of nodes and links in the network is normally 5 to 10 times bigger than in a comparable auto network. For this reason, obtaining a solution for a large transit network is challenging. This study will consider the application of the model for a relatively large transit schedule network.

- *Extension to the stochastic approach*

  We can also expand the proposed assignment models into stochastic assignment. The hyperpath-based assignment model on a transit schedule network can use logit-type proportions in the hyperpath search. Considering the stochastic assignment model introduced by Fisk (1980), the path-based assignment approach is typically used, since the entropy term in Fisk (1980) was created in terms of path flows (not link flows). Therefore, the path-based models allow us to explore a stochastic schedule-based transit assignment. In addition, the Independence of Irrelevant Alternatives (IIA) property of the logit route choice model, or the so-called "overlapping problem" on the searched paths, should be resolved effectively in this stochastic approach.

In addition, through this study, we also anticipate a critical contribution by providing efficient transit assignment models. As described in the previous chapter, the algorithms have been developed to solve the proposed problems, and the efficiency of the proposed heuristics in terms of computation performance will be considered in Chapter 10.

3 TRANSIT SCHEDULE NETWORK

3.1 Introduction

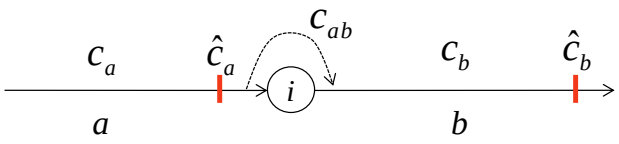
A path can be fundamentally represented by a series of nodes or links, typically with a label on each node or link. We call labeling on each node a "node-based" representation and on each link a "link-based" representation. A typical way to record a path is to keep a predecessor for each node or link. As a basic unit for a path, a node-based representation will keep a *node* unit in a predecessor set. On the other hand, a link-based representation will keep track of a *link* unit, or alternately, *node-to-node* as each link has a head node and a tail node. For this reason, when we consider a turning movement, the link-based representation maintains each turning movement without any network expansion. In contrast, the node-based representation requires additional links to capture turning movements.

This link-based representation is appropriate for characterizing transit supply and demand on a transit schedule network. *First*, transit supply can be easily represented on route-to-route or run-to-run with passenger transfers on the link-to-link scheme without extensive network expansion. Since the transfer behavior can be captured in a turning movement from one transit run to the other, a link-to-link turn penalty will give a concise and appropriate structure without the node expansion. *Second*, the priority of boarding is considered critical, especially when we consider transit vehicle congestion. The priority is extended in the same context of the link-to-link representation. The priority of boarding is strictly dependent on the transit schedule and transfers. *Third*, the link-to-link representation requires the number of links not more than the number of scheduled trip segments plus the transfer links. Since each link can connect two successive schedule points, the representation does not require having more fine temporal resolution like minute-by-minute representations like the time-expanded network of Hamdouch and Lawphongpanich (2008).

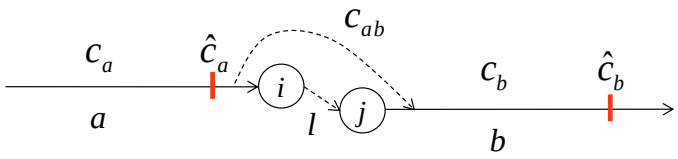
3.2 Time-Expanded Network

3.2.1 Link-Based Representation

Instead of updating a cost (or label) on each node, above all, the cost of each link is updated using the link-to-link scheme introduced by Potts and Oliver (1972), considering turn penalties in a transportation network. In Figure 3.1, link *a* and link *b* have their own link costs c_a and c_b, respectively. Every cost label is updated at the end of each link (i.e., $\hat{c}_a$ and $\hat{c}_b$), following Bellman's optimality rule, where c_{ab} is the turn penalty cost between two successive links, *a* and *b*. As in Figure 3.1 (a) and (b), a turn penalty can be assigned on any two links connected directly or indirectly. In Figure 3.1(b), any cost-related information between two nodes i and j, or neighboring stops, can be conveyed as a turn penalty between link *a* and *b*, which are indirectly connected by link *l*.



(a) Link-based cost update



(b) Transfer between two different nodes

Figure 3.1 Link-Based Cost Update

3.2.2 Link-Based Time-Expanded (LBTE) Transit Schedule Network

In a transit network, every stop has associated with it: (1) a sequence of points in time when a vehicle from a route will visit; and, (2) the travel time or the arrival time of the vehicle at the next available stop on the route after this stop. In one network representation, stops can be expanded based on points in time, and the time points are connected and expanded spatially by each bus run or route (Nuzzolo et al, 2001; Hamdouch and Lawphonpanich, 2008; Nielsen and Frederiksen, 2009). We also call this network an expanded node-based network, since the label is fundamentally updated through each node in a path search.

Instead of repeating the stop for each point in time, we propose that time points are assigned to each link connecting two stops by each run (or route). In this way, each link from a stop represents a run of each vehicle with departure time (the previous stop departure time of the run, τ_a^{dep}) and arrival time (the arrival time at the next stop for the run, τ_a^{arr}), as shown in Figure 3.2. The difference between the departure time and arrival time at the next stop is the travel time, $t_a^{trv} = \tau_a^{arr} - \tau_a^{dep}$, and any transfer cost including transfer or walking is defined by t_{ab}^{trsf}, and waiting time is defined by t_{ab}^{wait} for two consecutive links *a* and *b* and estimated to be $t_{ab}^{wait} = \tau_b^{dep} - \left(\tau_a^{arr} + t_{ab}^{trsf}\right)$. The proposed transit schedule network reduces the number of nodes and links in the time-expanded network, except for creating transfer links among different time points and runs (or routes).

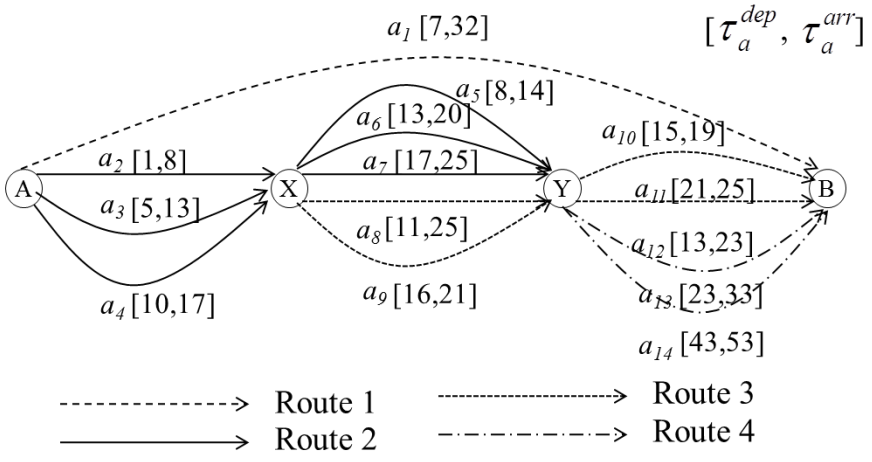
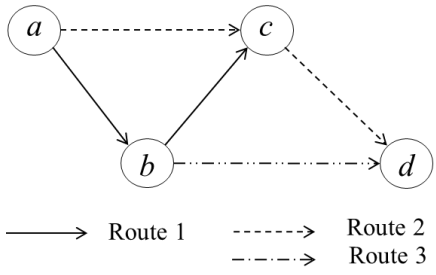
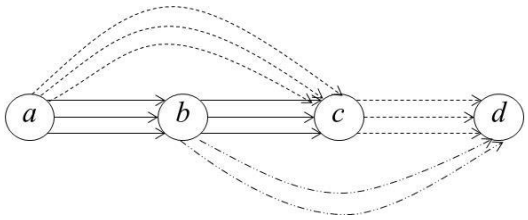


Figure 3.2 Link-Based Time-Expanded (LBTE) Transit Schedule Network (Noh et al., 2012a)

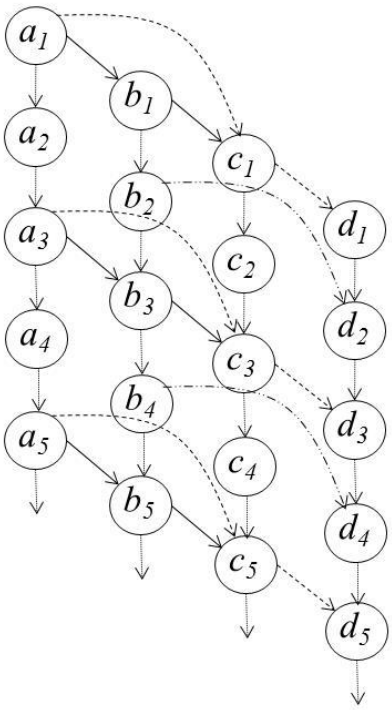
To compare the LBTE transit network with a node-based transit schedule network, let's assume a small transit network, used by Hamdouch and Lawphonpanich (2008), as shown in Figure 3.3(a) which consists of four nodes and three routes. We also assume that there are only three trips (or runs) for Routes 1 and 2 and two trips on Route 3.



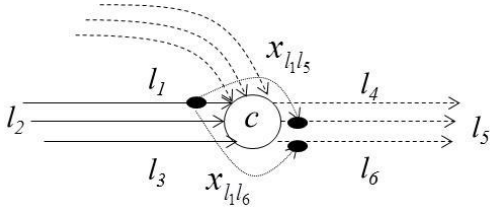
(a) Transit Network with Routes and Stops*



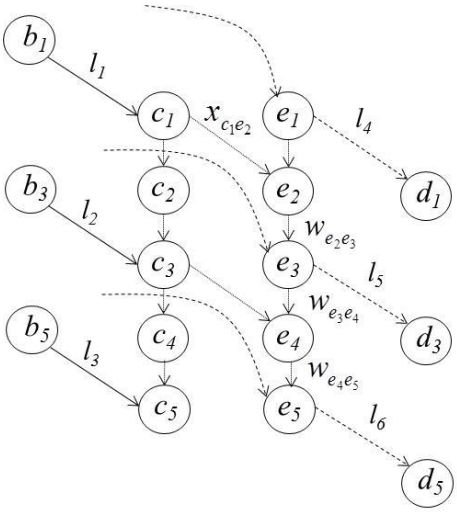
(b) LBTE Transit Network



(c) Node-expanded Transit Network *



(d) Transfer on LBTE Transit Network



(e) Transfer on Node-expanded Network

*: Used in (Hamdouch and Lawphonpanich, 2008)

Figure 3.3 Transit Schedule Network Expansion

For representing an appropriate transit passenger behavior, we consider two types of expansion as shown in Figure 3.3(b) and (c): the link- and node-expansion cases, respectively. *First*, as shown in Figure 3.3(c),

the node-expansion case will expand each stop or a representative node, aggregating neighboring stops to multiple temporal nodes. For example at stop *a*, there are available temporal nodes (a_1, a_2, a_3, a_4, and a_5) in which each node has a time component ($\tau_{a1}, \tau_{a2}, \tau_{a3}, \tau_{a4}, \text{and} \tau_{a5}$) and three trips depart from node a_1, a_3, and a_5 at time τ_{a1}, τ_{a3}, and τ_{a5}, respectively. The trips of Route 1 departing at node *a*, a_1, a_3, and a_5, are continued to nodes {b_1, c_1}, {b_3, c_3}, and {b_5, c_5}. And each node has one waiting link from one node to its successive downward node, like link (a_1,a_2). On the other hand, the link-based representation in Figure 3.3(b) expands links according to the route-level segment of a transit network, like Figure 3.3(a), in which a link has a departure time at the tail of link and an arrival time at the head of link, as shown in Figure 3.2. *Second*, when we consider transfers, there are major differences between link- and node-based representations, as shown in Figure 3.3(d) and (e), which are partial networks of Figure 3.3(b) and (c). Let's assume that node *c* in Figure 3.3(a) is considered for a transfer from Route 1 to Route 2, and we are interested in a transfer from trip l_1 to l_5 and l_1 to l_6 in Figure 3.3(d) and (e). In the node-based representation in Figure 3.3(e), the transfer from Route 1 to Route 2 requires an expanded set of nodes (e_1, e_2, e_3, e_4, and e_5). The transfer from l_1 to l_5 passes through transfer link $x_{c_1e_2}$ and waiting link $w_{e_2e_3}$, in which transfer time is assumed given and waiting time of $w_{e_2e_3}$ is estimated by the temporal length between τ_{e_2} and τ_{e_3}. The transfer from l_1 to l_6 is connected through the waiting links $w_{e_3e_4}$ and $w_{e_4e_5}$. On the other hand, in the LBTE network of Figure 3.3(d), the transfer link ($x_{l_1l_5}$) for the transfer from l_1 to l_5 contains transfer and arrival times, both where arrival time and waiting time are estimated by the alighting time at link l_1 and the transfer time to link l_5 , and the difference between the arrival time and scheduled departure time on link l_5 is as defined earlier. The transfer from l_1 to l_6 is represented by transfer link $x_{l_1l_6}$ in which waiting time is estimated in the same manner as mentioned earlier. This direct transfer link in the link-based representation allows fewer steps by skipping multiple temporal node-label updates, such as c_1, e_2, e_3, e_4, and e_5 for updating label between l_1 and l_6.

Considering the examples of Figure 3.3, these node- and link-based representations can be compared by the following. For the node expansion case, we consider a transit network G=(N,L) that has | N | nodes and | L | links according to Figure 3.3(a) in which each node i includes at least one physical stop. When each node i in a node set N has $|Q_i|$ scheduled runs, the stops in a transit schedule network will roughly expand to $\sum_{i=1}^{N}|Q_i|$ nodes. If each schedule is represented in a finer resolution like a minute-by-minute temporal representation, the total number of nodes will increase to $\sum_{i=1}^{N}|\overline{Q_i}| = \sum_{i=1}^{N}\gamma \cdot |Q_i|$ .where, γ is the average number of nodes between each node in its schedule time and the following node on its schedule time, such as node a_2 and a_4 in Figure 3.3(c) in which γ = 2. Since every node has at least one waiting link for boarding passengers, the number of links is estimated to be $\sum_{i=1}^{N}\delta \cdot |\overline{Q_i}|$ in which δ >1 for the connecting links, such as waiting and in-vehicle links; and, the last stop in a transit vehicle trip does not have a boarding link but waiting and transfer links are possible. We note that this estimate does not count access and egress links from the origin and to the destination. For transfer behavior, let's assume a transfer node set K. Each transfer node k in K has $|V_{in}^{k}|$ inbound and $|V_{out}^{k}|$ outbound routes and $|J^{k}|$ stops of transfer node k, where J^{k} is a set of stops at transfer node K. Here, we assume that each stop j in J^{k} only allows no more than one inbound and one outbound route, which means that each temporal node serves at most a single trip, such as node c_1 and e_1 serving only one trip of Routes 1 and 2 in Figure 3.3(e). Since each transfer node k has $|\overline{Q_k}|$ expanded nodes, the total transfer nodes are expanded to $|\overline{Q_k}| \cdot (|j^{k}| - 1)$, considering all stops in J^{k} but excluding the existing expanded nodes $|\overline{Q_k}|$ for each transfer node k. Because the transfer options are dependent on the arrival schedule of each transit trip at a specific stop, the number of transfer links of each stop j in J^{k} is also expanded to $|Q_j|$ with respect to the number of trips or $|\overline{Q_j}|$ only if we are interested in the high resolution of passenger transfer behavior,

like allowing a transfer from node c_2 to e_3 in Figure 3.3(e). But we only consider the normal case using $\left|Q_j\right|$ transfer links in this quantitative comparison. The total transfer links at a transfer node *k* is $\sum_{j=1}^{J^k}\left|Q_j\right|\cdot\left(\left|J^k\right|-1\right)$ since the transfer links are created to connect the current stop to the other stops *j*. Or, we may use $\sum_{j=1}^{J^k}\left|Q_j\right|\cdot\left(\left|V_{out}^k\right|-1\right)$ since the cardinality of outbound links are the same as the number of stops, as we assumed the in-degree and out-degree at each node. Finally, the total transfer links among | *K* | transfer nodes in a node-based transit schedule network will be $\sum_{k=1}^{K}\sum_{j=1}^{J^k}\left|Q_j\right|\cdot\left(\left|V_{out}^k\right|-1\right)$. Therefore, the total network in the node-based representation will be $\sum_{i=1}^{N}\left|\overline{Q}_i\right| + \sum_{k=1}^{K}(\left|j^k\right|-1)\cdot\left|\overline{Q}_k\right|$ nodes and $\sum_{i=1}^{N}\left|\overline{Q}_i\right|\cdot\delta + \sum_{k=1}^{K}\sum_{j=1}^{J^k}\left|Q_j\right|\cdot\left(\left|V_{out}^k\right|-1\right)$ links.

On the other hand, the number of nodes for the LBTE network is estimated to be | *N* |, since the network is not expanded by temporal nodes but by temporal links. The number of links is estimated to be $\sum_{i=1}^{N}\left|Q_i\right|$, in that links are generated only by the number of trips at each stop *i*. For estimating the number of transfers, let's revisit the example network, Figure 3.3(d). If we are only interested in the very next available link, like link l_5 from link l_1, the number of transfer links is simply estimated to be $\sum_{k=1}^{K}\sum_{j=1}^{J^k}\left|Q_j\right|\cdot\left(\left|V_{out}^k\right|-1\right)$, the same as the number of transfer links in the node-based expansion. However, if we are interested in multiple alternatives (i.e. $x_{l_1l_5}$ and $x_{l_1l_6}$ in Figure 3.3(d)), the number of transfer links is increased to $\sum_{k=1}^{K}\sum_{j=1}^{J^k}\varphi\cdot\left|Q_j\right|\cdot\left(\left|V_{out}^k\right|-1\right)$, where φ represents the number of interested alternatives ($\varphi\geq 1$), and will be increased by a wider temporal transfer window, i.e., 60 minutes from an alighting

stop to boarding on the next transit vehicle. Therefore, the total network size in the link-based representation is $|N|$ nodes and $\sum_{i=1}^{N}|Q_i| + \sum_{k=1}^{K}\sum_{j=1}^{J^k}\varphi\cdot|Q_j|\cdot\left(|V_{out}^k|-1\right)$ in-vehicle trip links and transfer links (including walking and waiting). Finally, since $|N| \le \sum_{i=1}^{N}|\overline{Q}_i| + (|j^k|-1)\cdot|\overline{Q}_k|$, the LBTE network uses many fewer nodes, and the number of links excluding transfer links is also competitive, since $\sum_{i=1}^{N}|Q_i| \le \sum_{i=1}^{N}|\overline{Q}_i|\cdot\delta$. On the other hand, the number of transfer links is dependent on the size of interesting alternatives, φ. If $\varphi=1$, $\sum_{i=1}^{N}|Q_i| + \sum_{k=1}^{K}\sum_{j=1}^{J^k}\varphi\cdot|Q_j|\cdot\left(|V_{out}^k|-1\right) \le \sum_{i=1}^{N}|\overline{Q}_i|\cdot\delta + \sum_{k=1}^{K}\sum_{j=1}^{J^k}|Q_j|\cdot\left(|V_{out}^k|-1\right)$, but if $\varphi>1$, it does not hold that the LBTE network representation has fewer links. However, as mentioned earlier, direct transfer links allow relatively fast cost label updates across links in the transfer, which allows the LBTE network to be competitive with a node-based transfer representation.

3.2.3 Acyclic LBTE Transit Schedule Network

A LBTE network consists of G(*S*,*A*), where *S* is a set of stops and *A* is the set of links. Each link $a \in A$ is the minimal unit of the network which includes starting and arriving stops (s^{dep}, s^{arr}) and times (τ^{dep}, τ^{arr}) as attributes, where *dep* and *arr* stand for departure and arrival on each link. The time attributes of the network can be eliminated, as this information is contained in an adjacency list holding that $b \in \overline{F}_a^+$ such that $\overline{F}_a^+ \equiv \{b|\ \tau_a^{arr} < \tau_b^{dep}\ \forall b \in F_a^+\}$. When we consider a turn penalty such as the transfer time in a transit network, it is also possible to represent G(*AB*), where AB is a set of link-to-link connectors. Including a turn penalty term, the adjacency list can be represented as $\overline{AB} \equiv \{ab|\ \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep}\ \forall a,b \in A\ \forall ab \in AB\}$. Then we can consider a link-to-link search with label updates by $\hat{c}_b = \min\left(\hat{c}_b, \hat{c}_a + c_{ab} + c_b\right)$ such that $a,b \in \overline{A} \subseteq A$ and $ab \in \overline{AB}$ when we consider a

turn penalty between two consecutive links *a* and $b \in \overline{F}_a^+$. The adjacency list consists that links with an earlier time cannot be searched from links arriving at a later time. The resulting network is acyclic, which can be proved in the following way. Assume a cycle that begins at link *a* and returns to link *a* through the adjacency list $\overline{AB}$, and that link *a'* is the predecessor link of link *a*, namely *a'a*. Each link *a* has τ_a^{dep} and τ_a^{arr} such that $\tau_a^{dep} < \tau_a^{arr}$. Link *a'a* will satisfy $\tau_a^{dep} < \tau_{a'}^{dep} = \tau_{a'}^{arr} + t_{a'a}^{trsf}$ because this is a cycle. However, since $\tau_{a'}^{dep} < \tau_{a'}^{arr} \le \tau_a^{dep}$ from the adjacency list, this contradicts the original assumption of a cyclic network.■

4 PATH MODELS AND ALGORITHMS ON A TRANSIT SCHEDULE NETWORK

With the proposed transit schedule network, the proposed path models are divided into two fundamental models. Utilizing the "link-based" structure (Potts and Oliver, 1972), a link-based shortest path (LBSP) and a link-based hyperpath (LBHP) model are introduced, and the properties of the models are investigated. We also consider the efficiency of the proposed path models when employing the hierarchical structure of transit schedules.

4.1 Shortest Path

4.1.1 Link-Based Shortest Path (LBSP)

On the proposed LBTE transit schedule network, we introduce two link-based shortest path models: a label-setting shortest path (LBSP) algorithm and a hierarchical LBSP algorithm. First, in Figure 4.1, the cost of link *a* is defined as $c_a = \beta^{trv} \cdot t_a^{trv}$, and the cost of link-to-link connector *ab* is $c_{ab} = c_{ab}^{trsf} + c_{ab}^{wait}$, where $c_{ab}^{trsf} = \beta^{trsf} \cdot t_{ab}^{trsf}$ and $c_{ab}^{wait} = \beta^{wait} \cdot t_{ab}^{wait}$, where β^{trsf}, β^{wait}, and β^{trv} are coefficients for the equivalent generalized "cost" for transfer, waiting, and travel time, respectively. The cost structure follows Bellman's principle of optimality, where link costs are additive and the label of link *b* is updated by $\hat{c}_b = \min\left(\hat{c}_b, \hat{c}_a + c_{ab} + c_b\right)$ in which $\hat{c}_b$ is the label of link *b* from origin *r*. Since all the time-related information on the network is static, we can use existing static shortest path models, such as Dijkstra's, with a link-based scheme (Ahuja et al. 1993).

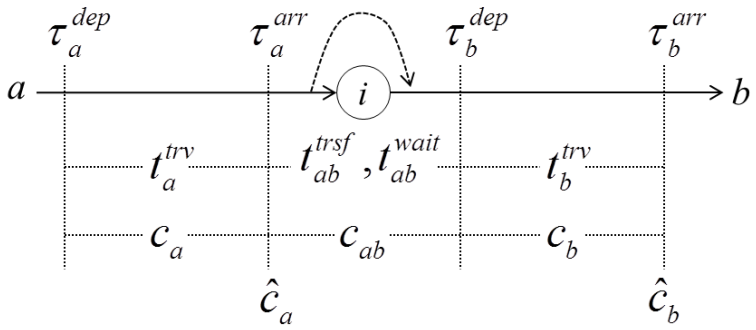


Figure 4.1 Cost Update on a LBTE Transit Network

4.1.2 Label-Setting LBSP (LS-LBSP)

Utilizing the adjacency list, which holds that an arrival link at a stop cannot connect to a link leaving the stop at an earlier time, a simple link-based shortest path algorithm using an array *Q* can be generated and is shown in Figure 4.2.

01: **for** ($a \in A, b \in B$)

02: $\overline{B}_b^- = \{a | \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep} \ \forall ab \in AB, \forall a,b \in A\}$ or $\overline{B}_b^- = \{a\} \ \forall ab \in \overline{AB}, \forall a,b \in A$;

03: $\overline{F}_a^+ = \{b | \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep} \ \forall ab \in AB, \forall a,b \in A\}$ or $\overline{F}_a^+ = \{b\} \ \forall ab \in \overline{AB}, \forall a,b \in A$;

04: $\hat{c}_a := \infty \ \forall a \in A$

05: $\hat{c}_s := 0$; $Q := Q \cup \{s\}$;

06: **while** ($Q \neq \emptyset$)

07: Retrieve arg-min $b \in Q$ and $Q := Q - \{b\}$;

08: **for** ($a \in \overline{B}_b^-$)

09: **if** ($\hat{c}_b + c_{ab} + c_a < \hat{c}_a$))

10: $\hat{c}_a = \hat{c}_b + c_{ab} + c_a$; $Q := Q \cup \{a\}$

Figure 4.2 Label-Setting LBSP (LS-LBSP) Algorithm

The time-dependent adjacency list is created through lines 01 to 03, generating a backward adjacency list $\overline{B}_b^-$ and a forward adjacency list $\overline{F}_b^+$. $ab \in \overline{AB}$ satisfies $\tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep} \ \forall a,b \in A, \forall ab \in AB$. The label of each link *a*, $\hat{c}_a \ \forall a \in A$, is initialized to infinity and in line 05, the label of the destination link *s* will be set to zero and the links to *s* are added to *Q*. The main loop is shown in steps 06 to 10. At lines 09 and 10, the algorithm maintains the optimality using Bellman's principle. The complexity of the algorithm is $\boldsymbol{O}(|A|^2)$ for the adjacency list in steps 01 and 02, and the main loop is also $\boldsymbol{O}(|A|^2)$ but if we utilize a heap structure on *Q*, the complexity is decreased to $\boldsymbol{O}\ (|A|\log|A|)$.

4.1.3 Hierarchical Representation on a Shortest Path

In a transit schedule network, a path model can be searched in a hierarchical manner. In this case, a subset of a scheduled run represents the basic unit for the path search, as introduced by Khani et al. (2012). To apply the model in the proposed link-based scheme, first, assume the optimal label at link *b*, $\hat{c}_b = \min\left(\hat{c}_b, c_b + \hat{c}_a + c_{ab}\right)$ and if $m_a = m_b$, where, m_a is a specific run or a trip using schedule link *a*, $\hat{c}_b = \min\left(\hat{c}_b, c_b + \hat{c}_a\right)$. In this case, transfer and waiting times are zero when m_a is the trip ID of link *a* and $|F_a^+| = 1$ for forward search and $|B_a^-| = 1$ for backward search. If a series of links, consisting of a subset of a path $(a_1, a_2, ..., a_n)$, satisfies $|F_{a_i}^+| = 1, i = 1, ..., n$ and $m_{a_1} = m_{a_2} = ... = m_{a_n}$, then the link label at link *a* is estimated by $\hat{c}_{a_n} = c_{a_1} + c_{a_2} + ... + c_{a_n} = \beta^{trv}\left(t_{a_1}^{trv} + t_{a_2}^{trv} + ... + t_{a_n}^{trv}\right) = \beta^{trv}(t_{a_n}^{arr} - t_{a_1}^{dep})$. Considering the property, we define a "leg" *l* as a series of links, $(a_1, a_2, ..., a_n)$, $l \in L$, where *L* is a set of legs. Then the transit schedule network is modified to $G(L, \overline{\overline{AB}})$, where $\overline{\overline{AB}}$ only consists of link-to-link connectors *ab* between (l_1, l_2) and the optimal label can be represented by $\hat{c}_{l_1} = \min\left(\hat{c}_{l_1}, \hat{c}_{l_2} + c_{ab} + c_{l_1} \mid ab \in \overline{\overline{AB}}, a = \tilde{h}(l_1), b = \tilde{t}(l_2)\right)$ where $\tilde{h}(l_1)$ is the head link of leg l_1 and $\tilde{t}(l_1)$ is the tail link of leg l_2.

4.1.4 Hierarchical Label-Setting LBSP (HLS-LBSP)

Assuming that transit schedules are static, the hierarchical approach is structured so as to keep schedule links only as they relate to passenger transfer behavior. When we have a predefined set of transfer links between two non-identical stops, each transit trip *T* can be broken to several consecutive legs $l \in L^T$ according to the possible transfer links. If each leg *l* has the information of the first link *a* and the last link *b*, the existing LBSP is applied on the transit schedule network. The hierarchical label-setting algorithm is shown in Figure 4.3.

01: **for** ($a \in A, b \in B$)

02: $\overline{F}_a^+ = \{b | \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep} \ \forall ab \in AB, \forall a,b \in A\}$ or $\overline{B}_b^- = \{a\} \ \forall ab \in \overline{AB}, \forall a,b \in A$;

03: $\overline{F}_a^+ = \{b | \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep} \ \forall ab \in AB, \forall a,b \in A\}$ or $\overline{F}_a^+ = \{b\} \ \forall ab \in \overline{AB}, \forall a,b \in A$;

04: $\hat{c}_a := \infty \ \forall a \in A$

05: $\hat{c}_s := 0$; $Q := Q \cup \{s\}$;

06: **while** ($Q \neq \emptyset$)

07: Retrieve arg-min $b \in Q$ and $Q := Q - \{b\}$;

08: **for** ($a \in \overline{B}_b^-$)

09: search l_a;

10: **if** ($\hat{c}_b + c_{ab} + c_{l_a} < \hat{c}_{l_a}$)

11: $\hat{c}_{l_a} = \hat{c}_b + c_{ab} + c_{l_a}$; $Q := Q \cup \{\tilde{t}(l_a)\}$

Figure 4.3 Hierarchical Label-Setting LBSP (HLS-LBSP) Algorithm

In the same way as the LBSP algorithm in Figure 4.2, lines 01 to 04 produce the adjacency list satisfying the time-dependent property and initialize a backward search by setting the label of the destination link *s*

to zero. The main loop has the same procedure from line 06 in Figure 4.2 except for line 09, which involves finding a link *a* preceding link *b* and in finding the leg *l* of the link *a*, in which all links in leg *l* have a same trip ID. In terms of performance, using heap structure on *Q*, the algorithm complexity is $\boldsymbol{O}(K \log |L|)$. The search of the tail link in a leg is assumed to take $\boldsymbol{O}(K)$ time in line 09. *K* is the possible number of alternatives at each link, defined in the interval $(1, |L|)$, where $|L|$ is the cardinality of leg *l*. Since $|L| \leq |A|$, the HLS-LBSP algorithm guarantees better performance than the LBSP algorithm.

4.2 Hyperpath

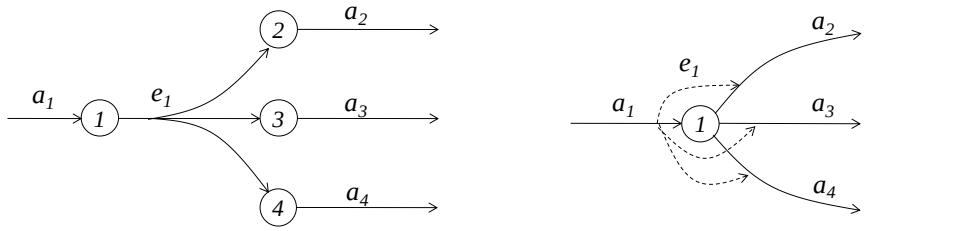
This section, adapted from Noh et al. (2012a), describes a hyperpath concept designed for a transit schedule network and details the corresponding algorithms for searching this hyperpath.

4.2.1 Definitions

The fundamental hyperpath model was introduced by Nguyen and Pallottino (1988) and Spiess and Florian (1989). With these initial studies, Gallo et al. (1993) defined a hyperpath in terms of generalized graph theory. According to Gallo et al. (1993), a *hyperlink* is defined to be $e = (t(e), h(e))$, where $t(e)$ is the tail node subset of hyperlink *e*, and $h(e)$ is the head node subset of hyperlink *e*. Each hyperlink is represented in the form of (node-link-node) in Gallo et al. (1993). Instead, we introduce a (link-to-link) hyperlink $e = (\{a\},\{b\})$, $\forall (a, b) \in AB$ and $a, b \in A$. We also define the hypergraph $H(A,E)$, where *E* is the set of hyperlinks. The difference between the proposed hypergraph and previous research is that the hyperlink connects two different link subsets, not two different node subsets. We define forward and backward link sets $F_a^+ = \{b \in A \setminus a \mid ab \in AB\}$ and $B_b^- = \{a \in A \setminus b \mid ab \in AB\}$, respectively. For hyperlink $e(\{a\},\{b\})$, if $|F_a^+| > 1$ and $|B_b^-| = 1$, then hyperlink *e* is diverging (one link leading to more than one other link), and merging (many links leading to a single other link) can be defined if $|F_a^+| = 1$

and $|B_b^-| > 1$. Otherwise, a simple monotonically-connected hyperlink (an "elementary" hyperlink) occurs when $|F_a^+| = 1$ and $|B_b^-| = 1$.

As an example, Figure 4.4 shows two possible representations of a hyperlink. Figure 4.4(a) represents a node-based hyperlink representation, while Figure 4.4(b) represents a link-based hyperlink representation.



(a) Node-based Diverging Hyperlink (b) Link-based Diverging Hyperlink

Figure 4.4 Diverging Hyperlink

In Figure 4.4(a), e_1 is the hyperlink using a node-based representation, where e_1 connects node 1 to a set of nodes {2, 3, 4}, which are subsequently connected to the nodes through links a_2, a_3, and a_4, respectively. On the other hand, in Figure 4.4(b), hyperlink e_1 connects a_1 to a_2, a_3, and a_4; that is, (a_1, $\{a_2, a_3, a_4\}$)$\in E$ such that (a_1,a_2), (a_1,a_3), (a_1,a_4) $\in AB$. Also, the possible hyperlinks (the dashed lines in Figure 4.4(b)) are provided by the combinations of a_1 and $\{a_2, a_3, a_4\}$. Another assumption in Figure 4.4(b) is that the hyperlink can be represented as separate connections to each link. If the network contains link costs and a weight function for the separate connections for each hyperlink, either a label-setting algorithm or a label-correcting algorithm can be used. For this reason, the hyperlink e in Figure 4.4(b) can be represented as F_a^+ or B_a^- $\forall a \in A$.

For the elementary hyperlink, the hyperlink cost is updated using the link-to-link scheme introduced by Potts and Oliver (1972), considering turn penalties in a transportation network. As mentioned in Chapter 3, when we assume two successive links *a* and link *b* which have their own link costs c_a and c_b, respectively, every cost label is updated at the end of each link (i.e., $\hat{c}_a$ and $\hat{c}_b$) following Bellman's optimality rule, $\hat{c}_b = \min\left(\hat{c}_b, \hat{c}_a + c_{ab} + c_b\right)$ and c_{ab} is the turn penalty cost. In the same manner, the hyperlink cost is updated using $\hat{c}_b = \min\left(\hat{c}_b, c_b + w\left(\left\{\hat{c}_a + c_{ab}\right\}\right) \mid a \in B_b^-\right)$ where $w(\cdot) = \min_{\{a\}} f\left(\left\{\hat{c}_a + c_{ab}\right\} \mid a \in B_b^-\right)$ and $f(\cdot)$ is the weighting function for the hyperlink. Therefore, Bellman's optimality rule is satisfied since $w(\cdot)$ is the minimum value on the hyperlink.

4.2.2 Proposed Hyperpath

The network conditions for a hyperpath were introduced by Nguyen and Pallottino (1988). These include: (1) the hyperpath h_{rs} is an acyclic hyperpath with at least one link connecting the origin *r* to the destination *s*; and, (2) at each node conditional probabilities for subsequent links exist, summing up to 1. These conditions were defined again by (Gallo et al., 1993; Nielsen et al., 2005). The proposed hyperpath in the LBTE network from origin *r* to destination *s* is formed with a series of links and hyperlinks as follows: $h_{rs} = \left(a_r, e_{a_r}, a_i, a_j, e_i, a_n, ..., e_l, a_s\right)$which can be represented using a forward link set for the diverging hyperlink case, with $h_{rs} = \left(a_r, a_i \in \overline{F}_{a_r}^+, a_j \in \overline{F}_{a_i}^+, ..., a_s \in \overline{F}_{a_k}^+, a_s \in \overline{F}_{a_l}^+\right)$ where $\overline{F}_a^+ \subseteq F_a^+$ $\forall a \in A$ and $r = h(a_r)$ and $s = t(a_s)$. The sub-hypergraph $\overline{H} = (\overline{A}, \overline{E}) = (\overline{A}, \overline{AB})$ assumes that a hyperlink can be separated into individual connections, such that (1) $\overline{A} \subseteq A$, (2) $a_r, a_s \in \overline{A}$, and (3) $\left|\overline{F}_a^+\right| \geq 1$ $\forall a \in \overline{A} \setminus \{a_s\}$ and $\left|\overline{B}_a^-\right| \geq 1$ $\forall a \in \overline{A} \setminus \{a_r\}$. As before, the bar over a variable means a subset of its original set. This definition of the hypergraph implicitly requires an acyclic network. In addition, the hypergraph can be represented by each origin link a_r, $\overline{H}_{a_r} \equiv \{b \mid b \in \widetilde{F}_a^+; \widetilde{F}_a^+ \subseteq \overline{F}_a^+, \forall a, b \in \overline{A}\}$

$\forall a_r \in \overline{A}$ or $\overline{H}_{a_r} \equiv \{b \mid (a,b) \in \overline{\overline{AB}}; \overline{\overline{AB}} \subseteq \overline{AB}, \forall a,b \in \overline{A}\}$ $\forall a_r \in \overline{A}$. In this case, we seek a hypergraph satisfying the optimality conditions with minimum weight, consisting of $\tilde{\overline{F}}_a^+$ for link *a* or $\overline{\overline{AB}}$ for (*a*,*b*), since the hypergraph is the union of elementary paths, with the connections *AB*.

4.2.3 Hyperpath Cost

When we consider a forward search from link *a* to *b*, the weight function on an LBTE network was defined earlier as $f\left(\left\{\hat{c}_a + c_{ab} \mid a \in B_b^-\right\}\right)$ with the weighting in $w_b(\cdot) = \min_{\{a\}} f\left(\left\{\hat{c}_a + c_{ab}\right\} \mid a \in B_b^-\right)$, where $\hat{c}_a$ is updated on the head of link *a*, *h*(*a*). On the other hand, a backward search from link *b* to *a* will need the waiting function, $w_a(\cdot) = \min_{\{b\}} f\left(\left\{\hat{c}_b + c_{ab}\right\} \mid b \in F_a^+\right)$. Of course, $\hat{c}_b$ is updated on the tail of link *b*, *t*(*b*). To prevent temporal violations and allow for a backward path search from the PAT at the destination, the weight function can be more clearly defined as $f\left(\left\{\hat{c}_a + c_{ab} \mid \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep},\, b \in F_a^+\right\}\right)$ and $w_a(\cdot) = \min_{\{b\}} f\left(\left\{\hat{c}_b + c_{ab} \mid \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep}\right\} \mid b \in F_a^+\right)$ or $w_a(\cdot) = \min_{\{b\}} f\left(\left\{\hat{c}_b + c_{ab}\right\} \mid b \in \overline{F}_a^+\right)$ as utilizing the adjacency list. Therefore, the cost at *h*(*a*) from the destination link a_s is $\hat{c}_a = \min\left(\hat{c}_a, c_a + w_a(\cdot) \mid b \in \overline{F}_a^+\right)$. It is worth noting that the weighting function may or may not satisfy the condition $w_a < \hat{c}_b + c_{ab}\ \forall b \in \overline{F}_a^+$.

The weighting function can be defined by several forms according to traveler behavior, most notably in the relationship between path costs and path alternatives. The available functions are: (i) an average model, in which the cost of all alternatives are averaged; (ii) a modified version of the optimal strategy by Spiess and Florian (1989) considering the transit schedule; and (iii) a log-sum model assuming stochastic user equilibrium (SUE) behavior. First, the benefit of using the average model is for the simplicity of the model, but it is limited in representing the behavior of transit passengers. When we utilize the average for the weighting function *f*, it is same as a shortest path since the optimal alternative set always chooses the

minimum cost alternative as $f(\{\hat{c}_a + c_{ab}\}) \geq \min\{\hat{c}_a + c_{ab}\}$ for searching a forward path. On the next alternative (a modified "optimal strategy"), the optimal set is possible to be configured by the relation between travel time and additional transfer and waiting time, similar to the optimal strategy suggested by Spiess and Florian (1989) as a deterministic weight function. The method brings the same sense of an optimal strategy; however, it is worth noting that a deterministic choice might lead passengers to choosing only the least cost alternative. Third, assuming that the perception of cost is different for each passenger, a log-sum weighting function will compensate their increased costs of more alternatives with the availability of more alternatives. For this reason, this third option is better than others when we consider a large number of alternatives and a possible change in cost depending on the number of alternatives. For this third case, the weight function can be represented by a log-sum function shown in Equation 4.1, where θ is the dispersion parameter for the logit model.

$$w_a = \min_{\{b\} \subseteq \bar{F}_a^+} \frac{1}{\theta} \ln \sum_b \exp(\theta \cdot \hat{c}_b) \quad \forall a \in A \tag{4.1}$$

The log-sum plays a role in the choice model for the transit schedule alternatives. The log-sum model has an issue that as more alternatives are added, no matter how high the cost is, the overall weight cost will decrease (or utility will increase). To manage this issue, the value of the dispersion parameter θ can be adjusted. It is also possible to reduce the number of alternatives by using a simple upper bound on the cost of alternatives in the alternative set. Based on the lowest cost alternative, the upper bound can be chosen by deciding the number of alternatives. Alternatively, we can also consider the logit probability of each alternative within the set of alternatives. The size of the set can be determined by allowing a certain minimum level of probability of a path, such as 0.0001.

In a transit schedule network, the link cost and weight could be generalized by including costs such as transfer time, waiting time and number of transfers, as shown in Equations 4.2 and 4.3.

$$w_a = \min_{\{b\} \subseteq \bar{F}_a^+} \frac{1}{\theta} \ln \sum_b \exp\left(\begin{matrix} \theta(\hat{c}_b + \beta^{trsf} t_{ab}^{trsf} + \beta^{wait} t_{ab}^{wait} \\ + \beta^{earlyPAT(or\,latePDT)} t_{ab}^{earlyPAT(or\ latePDT)} + \beta^{trsfNum} N^{trsf}) \end{matrix} \right) \quad \forall a \in A \qquad (4.2)$$

$$\hat{c}_a = \beta^{trv} t_a^{trv} + w_a \qquad (4.3)$$

where, t_a^{trv} is the (in-vehicle) travel time of link *a*; t_{ab}^{trsf} and t_{ab}^{wait} are transfer time and waiting time from link *a* to link *b*; $t_{ab}^{earlyPAT}$ and $t_{ab}^{latePDT}$ is the early arrival time relative to a preferred arrival time (PAT) and the late departure time relative to a preferred departure time (PDT), respectively; and N^{trsf} is number of transfers. The coefficients β^{trsf}, β^{wait}, $\beta^{earlyPAT}$, $\beta^{latePDT}$, $\beta^{trsfNum}$, and β^{trv} are parameters for transfer time, waiting time, relative early arrival and late departure time difference by PAT and PDT, transfer, and travel time, respectively. Transfer time is simply estimated by the distance from an alighting stop to the next boarding stop d_{ab}^{trsf} and passenger's walking speed $s^{walking}$; $t_{ab}^{trsf} = \frac{d_{ab}^{trsf}}{s^{walking}}$ and waiting time is estimated by $t_{ab}^{wait} = \tau_b^{dep} - \left(\tau_a^{arr} + t_{ab}^{trsf}\right)$ as shown in Figure 4.5, where, τ_b^{dep} and τ_a^{arr} stand for scheduled clock time departing from node *j* along link *b* and arriving to node *k* along link *a*.

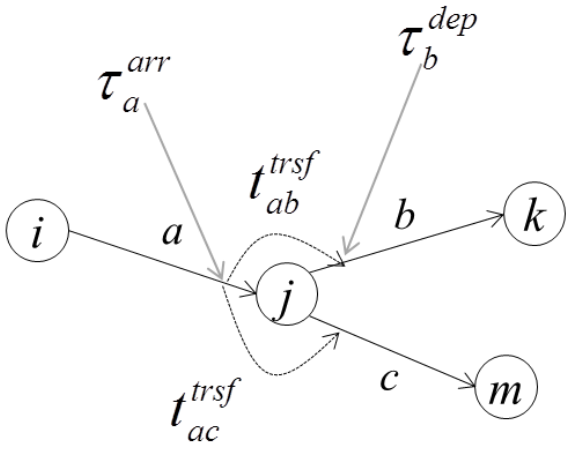
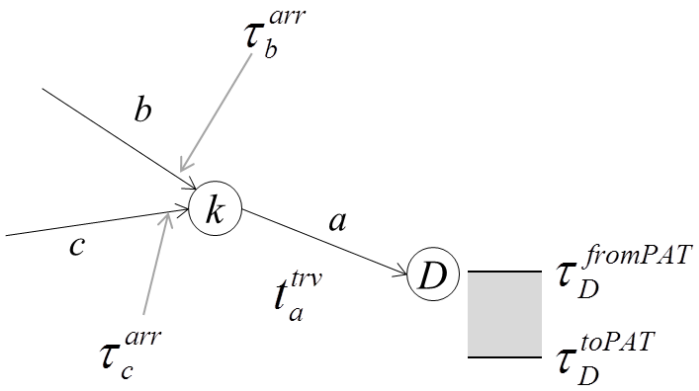


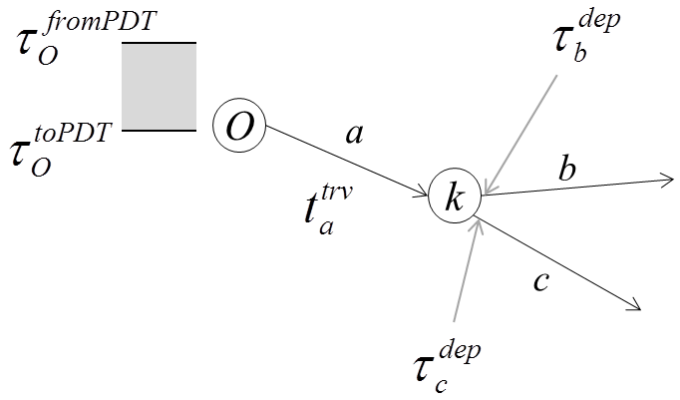
Figure 4.5 Transfer and Waiting Time

The daily traveler behavior is destined or originated either by PAT or PDT according to the purpose of the trips. Above all, PAT is representatively considered by destination-oriented activities such as work or school trips, typically in the morning peak hours. On the other hand, PDT is frequently applied for travel to origin-based activities usually associated with leaving from anchor activities such as work or school. PAT or PDT can have its own upper and lower time bounds consisting of a time buffer satisfying PAT or PDT. Figure 4.6 (a) and (b) show the PAT and PDT buffer and egress and access from/to transit links.

An early arrival time and a late departure time are estimated by the relative time difference from the latest arrival time at the destination for PAT, $t_{bD}^{EarlyPAT} = \tau_D^{latestArr} - \left(\tau_b^{arr} + t_a^{trv}\right)$, and from the earliest departure time at the origin for PDT, $t_{Ob}^{LatePDT} = \left(\tau_b^{dep} - t_a^{trv}\right) - \tau_O^{earliestDep}$, where $\tau_D^{latestArr}$ and $\tau_O^{earliestDep}$ stand for the latest arrival time at destination *D* among alternatives arriving to stop *k* and the earliest departure time from origin *O* among available transit alternatives departing from stop *k*. Instead of estimating the latest arrival time and the earliest departure time in advance for determining $t_{bD}^{EarlyPAT}$ and $t_{Ob}^{LatePDT}$, we may utilize the latest PAT, τ_D^{toPAT} and the earliest PDT $\tau_O^{fromPDT}$ to estimate the early arrival time and late departure time. Therefore, these early arrival and late departure times are estimated using $t_{bD}^{EarlyPAT} = \tau_D^{toPAT} - \left(\tau_b^{arr} + t_a^{trv}\right)$ and $t_{Ob}^{LatePDT} = \left(\tau_b^{dep} - t_a^{trv}\right) - \tau_O^{fromPDT}$ according to the alighting and boarding stops, respectively.



(a) Early Arrival Time



(b) Late Departure Time

Figure 4.6 Early Arrival and Late Departure Time

Along a backward or forward hyperpath by PAT or PDT (respectively), a log-sum weighting function conveys the travel time components as well as satisfying the bounded optimality condition. Since w_a satisfies the bounded optimality condition in Equation (4.2), $\hat{c}_a$ also maintains this optimality at the end of link *a* using Equation (4.3). Also, the log-sum weighting function carries all variables correctly over the conditional probability, similar to building a nested logit in the proposed hyperpath model.

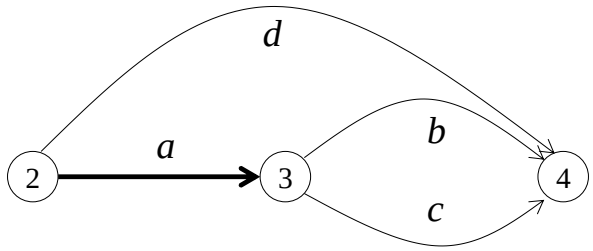


Figure 4.7 Hyperpath Cost UpdateUsing Nested Logit

This fact is explained through the following example. Let's consider a simple network as shown in Figure 4.7. The waiting cost for link *a* is updated by Equation (4.4) while the hyperpath is searched.

$$\hat{c}_a = \frac{1}{\theta}\ln\left[\exp\left(\theta \hat{c}_b\right) + \exp\left(\theta \hat{c}_c\right)\right] + c_a \tag{4.4}$$

When we define the probability of continuing from link *a* to link *b*, p_{ab}, this probability is simply a conditional probability, $p_{ab} = p_a p_{b|a}$ as follows.

$$p_a = \frac{\exp\left(\theta \hat{c}_a\right)}{\exp\left(\theta \hat{c}_a\right) + \exp\left(\theta \hat{c}_d\right)} = \frac{\exp\left(\theta c_a\right)\left[\exp\left(\theta \hat{c}_b\right) + \exp\left(\theta \hat{c}_c\right)\right]}{\exp\left(\theta c_a\right)\left[\exp\left(\theta \hat{c}_b\right) + \exp\left(\theta \hat{c}_c\right)\right] + \exp\left(\theta \hat{c}_d\right)} \tag{4.5}$$

$$p_{b|a} = \frac{\exp\left(\theta \hat{c}_b\right)}{\exp\left(\theta \hat{c}_b\right) + \exp\left(\theta \hat{c}_c\right)} \tag{4.6}$$

$$p_a p_{b|a} = \frac{\exp\left(\theta c_a + \theta c_b\right)}{\exp\left(\theta c_a + \theta c_b\right) + \exp\left(\theta c_a + \theta c_c\right) + \exp\left(\theta c_d\right)} \tag{4.7}$$

$\therefore\ p_a p_{b|a} = p_{ab}$

As a hyperpath incorporates this conditional probability in the log-sum calculation, it is also possible to deliver the total travel time through the transit schedules. Total travel time is just a byproduct when the log-sum weight function updates the link travel, transfer, and waiting times along a hyperpath. The log-sum calculation carries the current travel time to upper nests in the decision tree. Finally, this framework allows easy updating of the total travel time during a hyperpath search.

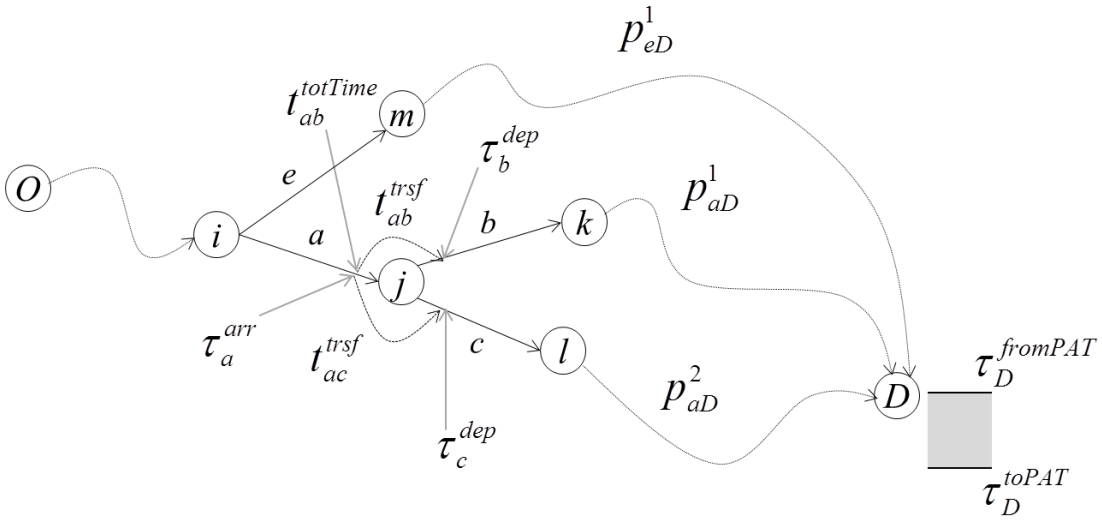


Figure 4.8 Travel Time Update on a Backward Hyperpath Search

In Figure 4.8, total travel times from link *a* to destination *D* through link *b* along path p_{aD}^1 is simply updated by the difference between the arrival time at link *a*, τ_a^{arr}, and the end of the PAT buffer, τ_D^{toPAT} (or $\tau_D^{fromPAT}$) which is represented in Equation (4.8). It is also possible to represent this travel time using the transfer and waiting time on descending link *b* as in Equation (4.9). This total travel time from link *a* to destination *D* is also embedded in the hyperpath using Equation (4.10), which separates the out-of-vehicle time (OVT) and the in-vehicle time (IVT).

$$t_{ab}^{totTime} = \tau_D^{fromPAT} - \tau_a^{arr} \tag{4.8}$$

$$= t_{ab}^{trsf} + t_{ab}^{wait} + \left(\tau_D^{fromPAT} - \tau_b^{dep}\right) \tag{4.9}$$

$$= \sum_{uv \in p_{aD}^1} \left(t_{uv}^{trsf} + t_{uv}^{wait}\right) + \sum_{u \in p_{aD}^1} t_u^{trv} = t_{p_{aD}^1}^{OVT} + t_{p_{aD}^1}^{IVT} \tag{4.10}$$

The same approach for updating the total travel time through a forward hyperpath is represented in Figure 4.9. Total travel time from link *a* to origin *O* (or from origin *O* to link *a*) is simply updated by Equation (4.11) as the difference between the end of the PDT buffer, τ_O^{toPDT} (or $\tau_O^{fromPDT}$) and the departure time on link *a*, τ_a^{dep}. The travel time representation is also expanded to use transfer, waiting and travel time or OVT and IVT as shown in Equations (4.12) and (4.13).

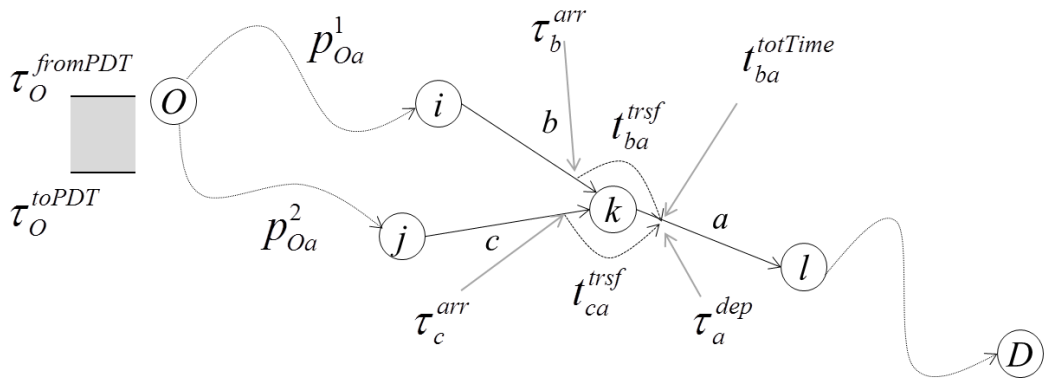


Figure 4.9 Travel Time Update on a Forward Hyperpath Search

$$t_{ba}^{totTime} = \tau_a^{dep} - \tau_O^{toPDT} \tag{4.11}$$

$$= t_{ba}^{trsf} + t_{ba}^{wait} + \left(\tau_b^{arr} - \tau_O^{toPDT}\right) \tag{4.12}$$

$$= \sum_{uv \in p_{Oa}^1} \left(t_{uv}^{trsf} + t_{uv}^{wait}\right) + \sum_{u \in p_{Oa}^1} t_u^{trv} = t_{p_{Oa}^1}^{OVT} + t_{p_{Oa}^1}^{IVT} \tag{4.13}$$

4.2.4 Label-Correcting LBHP (LC-LBHP)

To generate an optimal hyperpath on a LBTE transit network, we consider a label-correcting algorithm as shown in Figure 4.10.

01: **for** ($ab \in AB$)

02: $\overline{B}_b^- = \{a | \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep} \ \forall ab \in AB, \forall a,b \in A\}$ or $\overline{B}_b^- = \{a\} \ \forall ab \in \overline{AB}, \forall a,b \in \overline{A}$;

03: $\overline{F}_a^+ = \{b | \tau_a^{arr} + t_{ab}^{trsf} < \tau_b^{dep} \ \forall ab \in AB, \forall a,b \in A\}$ or $\overline{F}_a^+ = \{b\} \ \forall ab \in \overline{AB}, \forall a,b \in \overline{A}$;

04: $\hat{c}_a := \infty \ \forall a \in \overline{A}$

05: $\hat{c}_s := 0$;

06: **for** ($a \in \overline{B}_b^-$) $Q := Q \cup \{a\}$;

07: **while** ($Q \neq \emptyset$)

08: Retrieve $a \in Q$ and Q:=Q - {a};

09: **for** ($c \in \overline{B}_a^-$)

10: **if** ($\{c\} \cap Q = \emptyset$) **then** $Q := Q \cup \{c\}$

11: **for** ($b \in \overline{F}_a^+$)

12: $\hat{c}_a^{new}$:= ***alternative_set***(a,b);

13: **if** ($\hat{c}_a^{new} < \hat{c}_a$) $\hat{c}_a := \hat{c}_a^{new}$;

14: **if** ($\{a\} \cap Q = \emptyset$) **then** $Q := Q \cup \{a\}$

(a) Overall Algorithm

// ***alternative_set***(a, b)

15: **if** ($b = s$)

16: **if** ($\overline{H}_a \cap \{b\} = \emptyset$) $\overline{H}_a := \overline{H}_a \cup \{b\}$, and update $\hat{c}_a := c_b$;

17: **else**

18: **for** ($b \in \overline{F}_a^+$)

19: **if (** $\hat{c}_b < \hat{c}_{\bar{b}}$) $\bar{b} := b$; $\hat{c}_{\bar{b}} := \hat{c}_b$;

20: **for** ($b \in \overline{F}_a^+$)

21: $\hat{c}_b^{\exp} := \hat{c}_b^{\exp} + \exp(\hat{c}_b)$;

22: **if** ($\hat{c}_{\bar{b}} - \ln(\hat{c}_b^{\exp}) \le \beta$) $\hat{c}_{\bar{b}} = \ln(\hat{c}_b^{\exp})$; $\tilde{\overline{F}}_a^+ := \tilde{\overline{F}}_a^+ \cup \{b\}$;

23: $w_a = \min_{\{b\}} f\left(\left\{ \hat{c}_b + c_{ab} \right\} | b \in \overline{F}_a^+\right)$; $T_a = \arg\min_{\{b\}} f\left(\left\{ \hat{c}_b + c_{ab} \right\} | b \in \overline{F}_a^+\right)$;

24: **for** ($b \in T_a$)

25: **if** ($\overline{H}_a \cap \{b\} = \emptyset$) $\overline{H}_a := \overline{H}_a \cup \{b\}$;

26: **return** (w_a+c_a);

(b) Sub-algorithm for Optimal Set of Alternatives

Figure 4.10 Label-Correcting Hyperpath (LC-LBHP) Algorithm

We call this our *base* hyperpath algorithm. One distinctive characteristic of a hyperlink, say one leading link and its following alternative links, is that its cost on the leading link can be finalized only if the labels of all subsequent alternative linkss are updated. Also, the link-based scheme on a LBTE network allows a U-turn, so that an alternative for a link may not be finalized until all labels of alternative links, including the U-turn, are prepared. For these reasons, a generic label-setting algorithm may not end - there may be some un-scanned alternatives when the weighting cost of a hyperpath at the lead link is updated. In other words, a generic label-setting hyperpath search model may face an infeasible solution. For this reason, a

label-correcting algorithm is better to solve the hyperpath problem, although a relatively longer computation time is expected.

The proposed backward label-correcting algorithm on a LBTE schedule network is shown in Figure 4.10. The algorithm consists of a main function in Figure 4.10(a) and a sub-algorithm for determining the optimal set of alternatives in Figure 4.10(b). Initially, the main algorithm in Figure 4.10(a) generates the adjacency list satisfying the temporal constraints (lines 01 to 03), according to arrival and transfer times from any previous transit vehicle and the departure time of the next vehicle, similar to that shown in Figure 4.2 and Figure 4.3. After the initialization in lines 04 and 05, egress links are connected from the destination link and added to the search set Q in line 06. In the main loop, from lines 07 to 14, the labeling continues until Q is empty. The main loop has two sub-loops for building a hyperpath tree: (1) processes to add backward links from each processed link in lines 09 and 10; and, (2) defining the optimal set of alternatives in lines 11 to 14. For process (1), since a hyperlink defines the (link-to-link) relation, search set Q is expanded by adding preceding links. For each preceding link, process (2) creates a final set and updates the link cost. Then, if the link cost satisfies the optimality condition, the link with the new cost is added to Q. To generate the log-sum cost, we employ relative cost differences among alternatives, in comparison with the alternative with the minimum cost label.

The label-correcting algorithm has the complexity of $\boldsymbol{O}(D^2|A|(|A|+R))$, where D is the maximum number of alternatives connected from a leading (successor) link, $|A|$ is the total number of links, and R is the number of additional times one must revisit the same link in Q to correct the link's label. $(|A|+R)$ operations are taken within the **while** loop and $(D^2|A|)$ operations are taken in finding the set of alternatives, mainly dominated by the second **for** loop; see Figure 4.10(b). Since we utilize the label-correcting algorithm, R is a critical determinant of the algorithm's performance.

4.2.5 Label-Setting LBHP (LS-LBHP)

We can maximize the acyclic property on the LBTE network by maintaining the ascending order of schedule time for a backward search. By doing so, the label-correcting model transforms to a label-setting algorithm as shown in Figure 4.11. To understand this, assume that link *a* is the latest link received from *Q*. All the links *b* connected from link *a* on line 06 must have their labels, since the time-dependent acyclic property holds. This is why the label-setting hyperpath algorithm is effective, because we do not need to update the backward alternative links in line 09 and 10 in Figure 4.10(a). Also, the ***alternative_set*** (*a*,*b*) is the same as that in Figure 4.10(b).

01: $\hat{c}_a := \infty \ \forall a \in \overline{A}$

02: $\hat{c}_s := 0$;

03: **for** ($a \in \overline{B}_b^-$) $Q := Q \cup \{a\}$;

04: **while** ($Q \neq \emptyset$)

05: $\arg\max_{a \in Q}\{\tau_a^{arr}\}$ and Q:=Q - {a};

06: **for** ($b \in \overline{F}_a^+$)

07: $\hat{c}_a^{new}$:= ***alternative_set***(a,b);

08: **if** ($\hat{c}_a^{new} < \hat{c}_a$) $\hat{c}_{\bar{b}} := \hat{c}_b$;

09: **if** ({a}$\cap Q = \emptyset$) **then** $Q := Q \cup \{a\}$

Figure 4.11 Label-Setting Hyperpath Algorithm (Backward)

The complexity of the algorithm is $\boldsymbol{O}(D^2|A|^2)$, and it is possible to decrease this to $\boldsymbol{O}(D^2|A|\log|A|)$ when using a heap in *Q*. This is shown in Table 4.1, which also summarizes the complexity of other hyperpath models. The complexity of each model, including the proposed model, is categorized by "Outer-loop",

"Sub-algorithm", and "Overall". "Outer-loop" is the algorithm in Figure 4.10(a), excluding line 12, or in Figure 4.11, excluding line 07; and, "Sub-algorithm" stands for line 12 in Figure 4.10(b). "Overall" is the combination of "Outer-loop" with "Sub-algorithm". For "Outer-loop", $\boldsymbol{O}(|N|D^2)$ methods were produced by Gallo et al. (1993), Marcotte and Nguyen (1998), and Nielsen (2001), but Gallo et al. (1993) utilize heap sorting to obtain $\boldsymbol{O}(D^2 \log|N|)$. Other models by Nguyen and Pallottino (1989) and Nielsen (2001) show more simplified "Outer-loop" complexity of $\boldsymbol{O}(|N|D)$, typically as applied on an acyclic network, where, $|N|$ is the cardinality of nodes in the transit schedule network. Considering the link-based approach on an acyclic network, including heap sorting, the proposed model shows the complexity of $\boldsymbol{O}(D\log|A|)$. "Sub-algorithm" is generally a process to choose an optimal alternative set. As shown in Table 4.1, other models except for Nguyen and Pallottino (1989) simplify the process or assume that the set is given, where $\boldsymbol{O}(F)$ is the complexity of this simplified function or a given set. Since it is possible to generate a variety of "Sub-algorithms", it is difficult to choose the best model among them. However, if we assume that the "Sub-algorithm" uses a logit-type function, the proposed model is sufficiently competitive to other models using a link-based network representation.

Table 4.1 Complexity Comparison of Hyperpath Models

Model	Outer-loop	Sub-algorithm	Overall
Nguyen and Pallottino (1988): SHT	$\boldsymbol{O}(\|N\|D)$	$\boldsymbol{O}(D\log D)$	$\boldsymbol{O}(\|N\|D^2\log D)$
Gallo et al. (1993): SBT	$\boldsymbol{O}(D^2\log\|N\|)$	$\boldsymbol{O}(F)$	$\boldsymbol{O}(FD^2\log\|N\|)$
Nielsen (2001): SBT; Marcotte and Nguyen (1998)	$\boldsymbol{O}(\|N\|D^2)$	$\boldsymbol{O}(F)$	$\boldsymbol{O}(F\|N\|D^2)$
Nielsen (2001): SBT-acyclic	$\boldsymbol{O}(\|N\|D)$	$\boldsymbol{O}(F)$	$\boldsymbol{O}(F\|N\|D)$
Rochau et al. (2010): Backward pass	$\boldsymbol{O}(\|N\|D)$	$\boldsymbol{O}(F)$	$\boldsymbol{O}(F\|N\|D)$
Proposed Label-Setting Algorithm	$\boldsymbol{O}(D\log\|A\|)$	$\boldsymbol{O}(D\|A\|)$	$\boldsymbol{O}(D^2\|A\|\log\|A\|)$

Note: SHT = shortest hypertree; SBT = shortest b-tree; N= a set of node; A= a set of arc; D= maximum number of alternatives connected from a leading (successor) link; F = a simplified function, not explicitly explained in the algorithm

4.2.6 Hierarchical Representation on a Hypergraph

The hierarchical representation can be applied on a hypergraph in the same manner mentioned for the LBSP model in Chapter 4.1.4. The property of the hierarchical approach is explained in a hyperlink and optimality. The elementary hyperlink satisfying $\left|F_a^+\right|=1$ and $\left|B_b^-\right|=1$ holds that $\hat{c}_b=\min\left(\hat{c}_b, c_b+w\left(\left\{\hat{c}_b+c_{ab}\right\}\right)\right)=\min\left(\hat{c}_b, c_b+w\left(\hat{c}_b+c_{ab}\right)\right)$ and $w(\hat{c}_a+c_{ab})=\hat{c}_a+c_{ab}$. Therefore, $\hat{c}_b=\min\left(\hat{c}_b, c_b+\hat{c}_a+c_{ab}\right)$, and $\hat{c}_b=\min\left(\hat{c}_b, c_b+\hat{c}_a\right)$ if two consecutive trip IDs are the same, $m_a=m_b$ in which transfer and waiting times are zero. Also, a series of links consisting of a sub-path

satisfies the elementary hyperlink condition, $\hat{c}_{a_n} = \beta^{trv}(\tau^{arr}_{a_n} - \tau^{dep}_{a_1})$ such that $m_{a_1} = m_{a_2} = ... = m_{a_n}$.

Finally, if we have a subset of elementary hyperlinks, the travel time over that subset is estimated by the difference between the arrival time to the last link and the departure time from the first link.

For example, in Figure 4.12, a hyperpath will be searched from origin *r* to destination *s* along vehicle trips m_1, m_2, m_3, m_4, and m_5. Trip m_3 is divided into two subsets of links, which accommodate both the direct continuation on trip m_3 as well as a connection (transfer) from trip m_4. Also, trip m_1 and m_2 are connected by two access links from origin *r*, and trip m_3 and m_5 permit egress to the destination *s* using two egress links. Intuitively, a hyperpath is searched from 12 links (bold arrows) or 6 legs, connecting two successive arrows consisting of a head and a tail. Each leg is defined as a subset of links satisfying the elementary hyperlink condition mentioned earlier. This hyperpath connects from a_1 to a_{12}, excluding access and egress links, and transfer behavior only occurs between two different trips. The travel time of each leg is estimated by the departure time of the first link and the arrival time of the last link of each leg, such as $\left(\tau^{arr}_{a_{11}} - \tau^{dep}_{a_9}\right)$ on trip m_3.

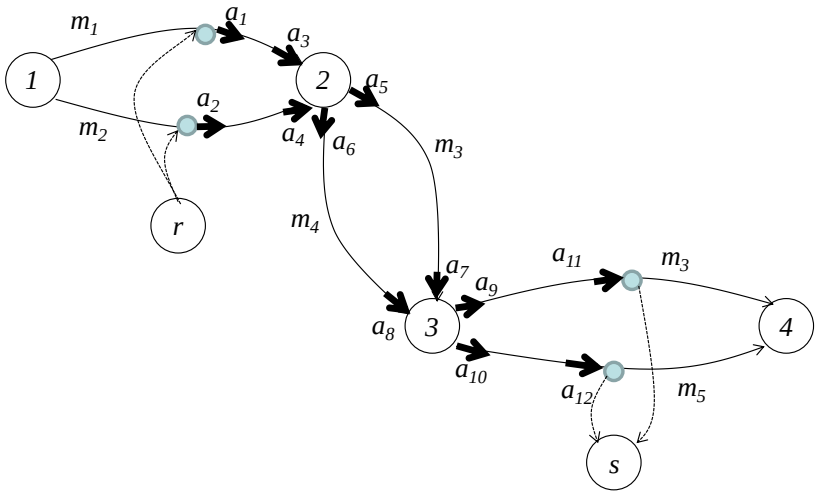


Figure 4.12 Hierarchical Hyperpath Representation on a LBTE Transit Network

In addition, we can consider a small change to improve the efficiency of the proposed hierarchical approach using the same network representation. When we assume a backward hyperpath search from a destination, it is possible to omit some origins without demand if we assume that all origin-destination pairs are known in advance. The links not relevant for the desired set of origin-destination pairs can be skipped, by defining each leg in advance according to the desired origin-destination pairs. By keeping the longer legs and omitting the unnecessary origins (or destinations for a forward search), the performance of a hyperpath search may be improved.

4.2.7 Hierarchical Label-Correcting and -Setting LBHP (HLC- and HLS-LBHP)

The hierarchical representation mentioned in the previous section can be applied to the hyperpath model by updating the cost of the first link on a leg tracked from the last link on the same leg or trip. In the proposed hyperpath labeling algorithm in Figure 4.10 and Figure 4.11, the hierarchical label-correcting approach is applied right after ***alternative_set***(*a*,*b*) in line 10 of Figure 4.13, since the weight on the hyperpath is typically updated when a transfer or a stop access occurs (with at least more than one in- or out-degree). On the leg *l*, link *a'* is the last link (or tail link) and *a* is the first link (or head link). By updating the label of link *a'* directly, instead of the label of link *a*, the other unnecessary intermediate links remain labeled at infinity. This hierarchical approach gives a complexity of $\boldsymbol{O}(D^2L(L+R))$ where L is a set of legs.

01: $\hat{c}_a := \infty \ \forall a \in \overline{A}$

02: $\hat{c}_s := 0$;

03: **for** ($a \in \overline{B}_b^-$) $Q := Q \cup \{a\}$;

04: **while** ($Q \neq \emptyset$)

05: Retrieve $a \in Q$ and $Q := Q - \{a\}$;

06: **for** ($c \in \overline{B}_a^-$)

07: **if** ($\{c\} \cap Q = \emptyset$) **then** $Q := Q \cup \{c\}$

08: **for** ($b \in \overline{F}_a^+$)

09: $\hat{c}_a^{new}$:= ***alternative_set*** (a,b);

10: search l_a and $\hat{c}_{a'}^{new} := \hat{c}_a^{new} + c_{l_a}$;

11: **if** ($\hat{c}_{a'}^{new} < \hat{c}_{a'}$) $\hat{c}_{a'} := \hat{c}_{a'}^{new}$;

12: **if** ($\{a'\} \cap Q = \emptyset$) **then** $Q := Q \cup \{a'\}$

Figure 4.13 Hierarchical Label-Correcting Hyperpath (HLC-LBHP) Algorithm

When the hierarchical scheme is applied using a label-setting hyperpath model, the algorithm is not only simpler but also does perform better, as shown in Figure 4.14. In line 07, the same approach is applied in referring link a' for the label update. Finally, the algorithm generates a complexity of $\boldsymbol{O}(D^2|L| \log |L|)$.

01: $\hat{c}_s := 0 \ \forall s \in \overline{A}$;

02: **for** ($a \in \overline{B}_b^-$) $Q := Q \cup \{a\}$;

03: **while** ($Q \neq \emptyset$)

04: $\arg\max_{a \in Q}\{\tau_a^{arr}\}$ and Q:=Q - {a};

05: **for** ($b \in \overline{F}_a^+$)

06: $\hat{c}_a^{new}$:= ***alternative_set***(a,b);

07: search l_a (a) and $\hat{c}_{a'}^{new} := \hat{c}_a^{new} + c_{l_a}$;

08: **if** ($\hat{c}_{a'}^{new} < \hat{c}_{a'}$) $\hat{c}_{a'} := \hat{c}_{a'}^{new}$;

09: **if** ({a'}$\cap Q$ = $\emptyset$) **then** Q := $Q \cup$ {a'}

Figure 4.14 Hierarchical Label-Setting Hyperpath (HLS-LBHP) Algorithm

5 TRANSIT ASSIGNMENT ON A LBTE SCHEDULE NETWORK

5.1 Introduction

The necessary ingredient of a transportation assignment model is an efficient path from origin to destination. In the previous chapter, we explored several transit path algorithms and their performances, which are typically compatible with a LBTE transit schedule network. In terms of applying these various path algorithms to transit assignment, another interesting and challenging task in this study resides in how to determine a feasible solution including the transit vehicle capacity constraint. The capacity problem over a transit schedule network requests a thoughtful understanding of passenger behavior. To find out proper methodologies to solve this transit capacity problem, we may look over existing assignment models. Fortunately, models and techniques solving the capacity problem have been studied widely in the auto assignment area. Although it is genetically different to the capacity problem in transit assignment, it is good to explore existing models and techniques in the same context of finding a better path to minimize the travel cost of each passenger. In terms of providing a set of feasible paths efficiently under the capacity constraint, in this study, we are interested in path-based assignment models on a LBTE transit schedule network as well as finding another model utilizing a hyperpath algorithm.

Before applying the existing assignment models, *first*, the relation between passengers' movement and vehicle capacity should be thoroughly understood within the user equilibrium (UE) on a transit schedule network, and a proper quantitative capacity constraint should be posed with adequate assumptions. On the proposed capacity model, we also need to think about how to solve the problem efficiently, in terms of getting a feasible solution.

To this end, *second*, three assignment models are introduced on the following chapters: (1) a hyperpath-based model using the method of successive averages (MSA); and, two path-based models using (2) the

gradient projection and (3) self-adaptive gradient projection for solving the deterministic and stochastic user equilibrium.

5.2 Behavioral Assumptions, User Equilibrium, and Initial Feasible Solution

5.2.1 Fundamental Behavioral Assumptions

Above all, in terms of a schedule-based approach, passengers are assumed to respond the fixed schedule. According to the schedule, each transit vehicle arrives and departs on time, and the number of passengers are assumed not to affect the waiting time so that the marginal boarding time and alighting time are assumed to be zero. Second, for the travel between an origin and a destination, a passenger will have a preferred arrival time (PAT) at the destination or a preferred departure time (PDT) at the origin. Third, a passenger will minimize his/her total expected cost which is represented in a generalized cost term. Fourth, the passenger behaves as an individual (no groups), and all passengers are assumed to have identical behavior in the deterministic UE models and different perceptions of cost in the stochastic UE models. Fifth, regarding boarding at stops and vehicle capacity, sending flows that exceed the given vehicle capacity is allowed, but this capacity violation will be treated with a penalty cost. By allowing this violation, we call this relaxation a "soft" capacity model, typically using a monotonically increasing penalty function dependent on the number of boarding passengers and the residual capacity on board the vehicle.

5.2.2 Passenger Priority

One distinctive difference of the passenger behavior from auto (driver) behavior is in the priority of boarding. Each passenger will have his/her priority for boarding according to his/her arrival time at a stop. Hamdouch and Lawphonpanich (2008, 2010) and Poon et al. (2004) mentioned this FIFO priority.

Therefore, on a transit schedule network, passengers already on board have the first priority, then the passengers arriving to the stop will have priority directly related to their time of arrival. This priority behavior is represented on a LBTE transit schedule network as follows.

$$o_{ab}^{m_a m_b} < o_{a_i b}^{m_{a_i} m_b} \; if \;\; m_a = m_b, m_{a_i} \neq m_b, a_i \neq a, \{a_i\} \cup a = B_b^- \tag{5.1}$$

$$o_{ab}^{m_a m_b} < o_{a_i b}^{m_{a_i} m_b} \; if \;\; m_a \neq m_b, m_{a_i} \neq m_b, \left(\tau_a^{arr} + t_{ab}^{trsf}\right) < \left(\tau_{a_i}^{arr} + t_{a_i b}^{trsf}\right) \;\; a_i \neq a, \{a_i\} \cup a = B_b^- \tag{5.2}$$

where, $o_{ab}^{m_a m_b}$ is the order of priority on the link-to-link connector (or transfer connector) *ab*, connecting trip m_a to m_b. Among the alternatives in the backward link set of link *b*, the alternative link *a* with the same vehicle run of link *b* will have the first priority, as shown in Equation (5.1). The second priority in Equation (5.2) (5.2) is activated by the passengers' arrival time to link *b*, $\tau_a^{arr} + t_{ab}^{trsf}$, such that the earlier arrival time, either directly to the stop or in the transfer, has the higher priority.

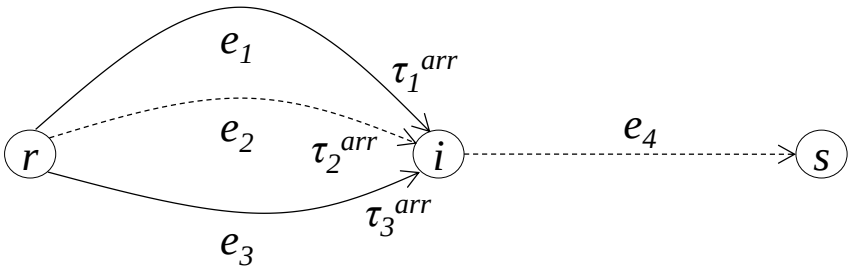


Figure 5.1 Passenger Priority Representation

In Figure 5.1, we assume that three links (e_1, e_2, and e_3), with arrival times ($\tau_1^{arr} < \tau_2^{arr} < \tau_3^{arr}$) at node *i*, are connected to link e_4 and the transit vehicle runs of e_2 connect directly to e_4 (the dashed lines), which includes $t_{e2e4}^{trsf} = 0$, $t_{e1e4}^{trsf} > 0$, and $t_{e3e4}^{trsf} > 0$. The passengers on link e_2 will have the first priority to

proceed to link e_4 since they are already on board. The second and third priority is ordered by the arrival time, $\tau_a^{arr} + t_{ab}^{trsf}$ of each vehicle from link e_1 and e_3. This order of priority will then be used to manage the boarding process when there is limited transit vehicle capacity.

5.2.3 Capacitated User Equilibrium (UE) on a Transit Schedule Network

In a congested transit network, a typical example is shown in Figure 5.2, which is a capacitated UE problem given by Nguyen et al. (2001). Assume that the network has three different routes, that 21 passengers will be assigned to the OD pair, and that each link has its cost and capacity. The anticipated system optimal (SO) result is depicted in Figure 5.2(a). Every passenger will use the upper-most route, up to its capacity, and a single generous passenger will choose the bottom route to reduce the total network cost. However, if the passenger wants to reduce his/her disadvantage, the passenger will use the second (middle) route, and all passengers will then shift to the second route until no more capacity is left. After all, the main reason for moving from path (a) to path (b) is the path priority. The second route dominates the first (upper) route, because it has priority over the upper link to proceed onto link (*i*,D).

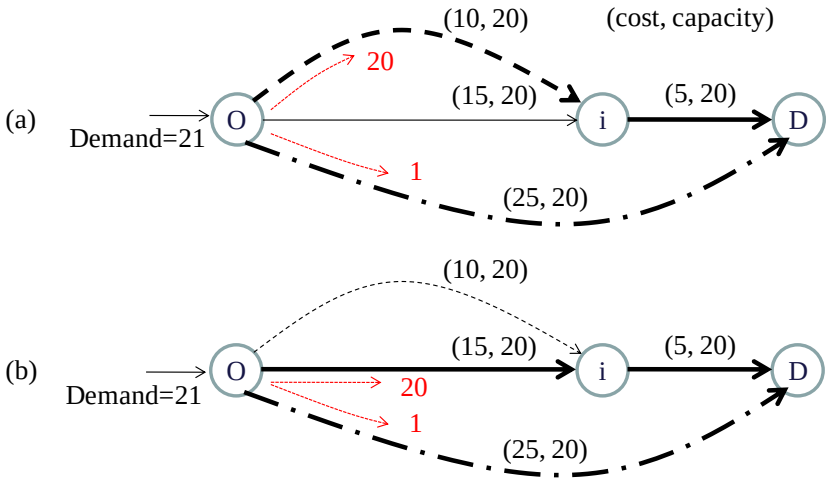


Figure 5.2 (a) SO and (b) UE on a Capacitated Transit Network

For this problem, Nguyen et al. (2001) considered the priority of the route to apply a penalty cost for a transfer, in order to generate a deterministic UE. The chosen penalty cost function in that study was a type of an exponential function, although they also mentioned the difficulty of practical applications with this type of penalty cost. By extending a similar concept to a logit-based hyperpath in a transit schedule network, we can consider the capacitated SUE problem. Also, this capacity penalty approach with passenger priority can be extended to a path-based assignment approach.

5.2.4 Vehicle Capacity and Capacity Penalty Function

With priority in the boarding process, congestion will be handled with a capacity constraint, captured through a boarding penalty function associated with transfer and waiting times, t_{ab}^{trsf} and t_{ab}^{wait}. According to the arrival time at a forward link *b* from link *a*, passengers will be loaded on the vehicle at a specific time. To include a "cost" for capacity violations, we consider two possible functions. As shown in Equations (5.3) and (5.4), the residual capacity r_b is reduced by the other priority flows. As the residual capacity of link *b* is reduced, the slope of the capacity penalty function increases monotonically and will become infinite as it approaches zero residual capacity. A similar type of penalty function was introduced by Nguyen et al. (2001). In Equations (5.3) and (5.4), the mathematical form of the capacity penalty cost is assumed to follow an exponential curve and a power curve, where α and β are parameters for determining the steepness of the capacity cost function and the sensitivity to residual capacity.

$$c_{ab}^{cap} = \frac{f_{ab}}{\max[0, r_b - \sum_{o_{kb}^{m_k m_b} < o_{ab}^{m_a m_b}} f_{kb}]} \exp\left(\alpha \cdot (f_{ab} - \max[0, r_b - \sum_{o_{kb}^{m_k m_b} < o_{ab}^{m_a m_b}} f_{kb}])\right) \tag{5.3}$$

$$c_{ab}^{cap} = \frac{f_{ab}}{\max[0, r_b - \sum_{o_{kb}^{m_k m_b} < o_{ab}^{m_a m_b}} f_{kb}]} \max\left[0, f_{ab} - \beta \cdot \max[0, r_b - \sum_{o_{kb}^{m_k m_b} < o_{ab}^{m_a m_b}} f_{kb}]\right]^{\alpha} \tag{5.4}$$

In the $\max[\bullet]^{\alpha}$ for the penalty functions Equations (5.3) and (5.4), the capacity penalty is maintained at zero until the assigned flows f_{ab} , or the sum of higher priority flows $\sum_{o_{kb}^{m_k m_b} < o_{ab}^{m_a m_b}} f_{kb}$ or $\sum_{k \in \text{Priority}(a)} f_{kb}$, exceed the residual capacity r_b. The residual capacity r_b in the denominator will play a role to push more flows to the other available routes when the residual capacity r_b is close to zero, because the function assigns a high penalty for the flows transferring to link *b*.

When we apply the proposed penalty function, reducing the residual capacity to zero can make the assignment model infeasible. For this reason, in the denominator of the penalty function Equation (5.3), $\max[\bullet]$ is simplified to r_{ab} as shown in Equation (5.5), and we set the lower bound $\underline{r_{ab}}$ of r_{ab} to a very small value, such as 0.001. Equation (5.6) shows the capacity cost function with the lower bound $\underline{r_{ab}}$.

$$r_{ab} = \max[\,0, r_b - \sum_{k \in \text{Priority}(a)} f_{kb}\,] \tag{5.5}$$

$$c_{ab} = \begin{cases} \dfrac{f_{ab}}{\underline{r_{ab}}} e^{\alpha(f_{ab} - \underline{r_{ab}})} & \text{if } r_{ab} = 0 \\ \dfrac{f_{ab}}{r_{ab}} e^{\alpha(f_{ab} - r_{ab})} & \text{if } r_{ab} > 0 \end{cases} \tag{5.6}$$

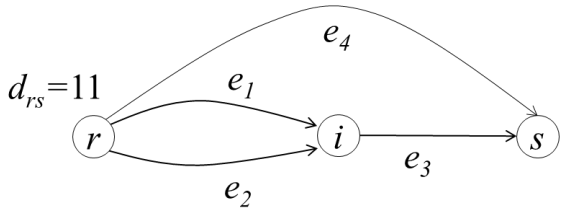
When we apply the penalty functions from Equations (5.3) and (5.4), we also need to consider one more critical element in dealing with the penalty cost: a variable residual capacity could make the assignment model unstable. Once flows with priority are assigned on a forward link, the residual capacity of the forward link will be reduced by these flows. Then, other flows with lower priority proceed onto the forward link, but face a lower (modified) residual capacity. Since the flows are endogenous in the assignment method, the resulting penalty costs can change within the assignment. To avoid sudden

changes in residual capacity, *first,* we use a diagonalization technique proposed by Sheffi (1985). For a transit schedule network, each link's schedule and transfer times are assumed to be unchanged by any other passenger flows. This assumption creates a fixed priority on the schedule network for every boarding case. In each diagonalization stage, the residual capacity is fixed until the UE or SUE conditions are satisfied. Then, the flows, the residual capacities, and the penalty costs are updated sequentially for the next diagonalization iteration. *Second,* instead of fixing the vehicle capacity by using a diagonalization technique with an asymmetric cost relation, we can also consider any quadratic models like a gradient projection, because of the strong FIFO priority of boarding. This FIFO priority orders the boarding priorities among the alternatives at each node, and the capacity cost is only dependent on the higher order alternatives. In other words, because the capacity cost depends on the boarding priority, the Jacobian of the cost function is an upper or lower triangular matrix, ensuring the Jacobian is positive semi-definite. Finally, it allows an existing quadratic model using scaling matrix like Hessian matrix. However, it may have a counter example if path *A* has priority over path *B* at one transfer location, but path *B* to have priority over path *A.* In this case, since the capacity costs on both path *A* and *B* are affecting each other which allows a non-triangluar form, it may hinder the flow shift between the paths and the computational performance.

5.2.5 Better Initial Solution (BIS) Method

5.2.5.1 Shortcomings on Diagonalization Technique

Regarding the asymmetric cost and the priority of boarding in transit assignment, the diagonalization technique may make the performance of the proposed solution method relatively slow, because of the numerous diagonalization iterations to obtain convergence. For example, we have a simple network in Figure 5.3. Four links are connected from origin *r* to destination *s* with a demand of 11. Every link cost and capacity is shown in the table of Figure 5.3. For the priority, we assume that the path (e_2, e_3) has the priority on link e_3 , compared with the pagainst path (e_1, e_3).



Link	cost	capacity
e1	3	∞
e2	4	∞
e3	3	10
e4	9	∞

Figure 5.3 Example Network for Diagonalization in Transit Assignment

The UE solution sends 10 units of flow on the priority path (e_2, e_3) and one unit of flow on path (e_4). To solve the problem, we may start with an all-or-nothing loading on the shortest path (e_1, e_3). By violating the capacity on link e_3 ,sending a flow of 11 on the shortest path, a capacity cost will be assigned on link (e_1, e_3). Then, a flow adjustment along the paths will be continued until reaching a local UE solution in the diagonalization step. In this step we note that the residual capacity for estimating the capacity cost on link e_3 is fixed to 10 units of flow for both path (e_2, e_3) and path (e_1, e_3) until the assignment reaches the local equilibrium. Then, the residual capacities for the alternative paths will be updated. In the next iteration, the capacity of path (e_2, e_3) remains 10 but the capacity of path (e_1, e_3) is reduced by the lower priority of boarding on link e_3. As the diagonalization iteration goes on, the residual capacity on path (e_1, e_3) will be gradually diminished and finally eliminated, as link e_3 is saturated with the flows on path (e_2, e_3). Instead, if we started with the residual capacity of 0 on path (e_1, e_3) since we know the congested link e_3 and the priority of alternative link e_2, we may not need the additional diagonalization iterations when the equilibrated flows on path (e_2, e_3) increase from 0 to 10.

5.2.5.2 Deterministic Better Initial Solution (D-BIS)

To improve the performance of the proposed model, an initial feasible solution may more carefully consider the priority of boarding and the vehicle capacity on a congested link. As mentioned in the previous paragraph, if we set residual capacities to 10 and 0 for path (e_2, e_3) and path (e_1, e_3) respectively,

and send all 10 units of flow first on path (e_2, e_3), the performance of the algorithm will be improved by skipping these unnecessary diagonalization iterations. In terms of getting a better initial solution, we may *first* think of modifying the solution methodology. A min-cost flow problem is one approach to achieve a better initial solution, especially considering the capacitated links on the LBTE transit schedule network. A heuristic pseudo algorithm is shown in Figure 5.4.

Step 1: Find the shortest path (SP) and load all flows over the given O-D pairs

Step 2: Detect congested links

Step 3: According to the priority, assign the maximum capacity penalty

Step 4: Min-Cost Flow problem (i.e., Successive Shortest Path) with the penalty cost heuristically.

Step 4.0: Initialize link potential $\pi_a := 0$ and reduced cost $c_{ab}^{\pi} := c_{ab} + c_b$

Step 4.1: Find SP$^{\pi}$ using the reduced cost and update $\pi_b^{new} = \pi_b - d$ and c_{ab}^{π}

Step 4.2: Augment flows along SP$^{\pi}$ ($c_{ab}^{\pi} = 0$).

Step 4.3: Update residual capacity

Figure 5.4 Min-cost Flow Problem with Capacity

To apply this pseudo algorithm, it is necessary to redefine the potential and the reduced cost accompanying the LBTE transit network, including the link-to-link turning penalties. Before explaining the proposed algorithm in Figure 5.4, we consider two simple links shown in Figure 5.5. Instead of placing each potential on node *i*, we put it on the head and the tail of each link, such as $\pi_{b'}$ and π_b on

link *b*, where *d* represents the shortest distance to link *b* in Figure 5.4. This reduced cost is simply updated by these two successive link potentials, such as $c_{ab}^{\pi} = c_{ab} - \pi_a + \pi_{b'}$ and $c_{b'b}^{\pi} = c_b - \pi_{b'} + \pi_b$. The modified reduced cost will be maintained in a LBTE transit network. The optimality of non-negative reduced cost for the final flows x^*, $c_{ab}^{\pi}(x^*) \geq 0$, holds in this modification (Ahuja et al. 1993).

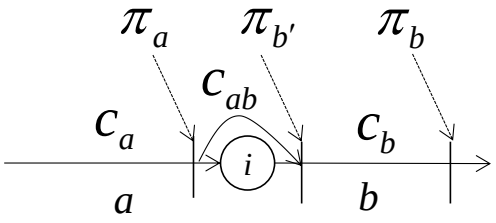


Figure 5.5 Link Potential and Reduced Cost

When we apply this algorithm, several challenging tasks reside on the residual capacity update in Step 4.3 associated with (1) adding and deleting each link from the adjacency list used in the shortest path and shortest hyperpath algorithms in Chapter 4, and (2) connecting to the main assignment module with the final feasible solution from this algorithm. Also, we note that the optimality of this algorithm is only guaranteed for specific capacity constraints by Step 3, so this requires having additional algorithm runs to reach the capacitated deterministic UE solution. For example, the penalty cost in Step 3 penalizes some movement link *a* to *b* which may have some positive residual capacity for other flows. In the following iteration, after updating new residual capacity according to the previous solution, it solves a new min-cost flow problem for the left flows with the positive residual capacity. This process continues until no more flow is left.

Instead of having this major network modification with the existing adjacency list, we may think about a heuristic method in terms of getting a better initial solution rather than reaching an optimal solution.

Second, the next possible solution model directly and heuristically controls the flows associated with the priority of boarding on congested links. To solve the problem, we suggest a bush-type heuristic shown in Figure 5.6.

Step 0: Search a congested link according to time order and/or proximity to destination

Step 1: Find a priority path using (origin, destination, time, tour, and trip-trip)

Step 2: Convert flows to the priority path until flows reach capacity

Step 3: Send remaining flows along the next available paths

Figure 5.6 Bush-type Heuristic Algorithm

Above all, path enumeration and link-path mapping are assumed for applying the heuristic in Figure 5.6. The proposed heuristic is simpler than the previous min-cost flow algorithm in terms of utilizing existing resources. In Step 2, the algorithm sends the flows along the priority path (Step 1), saturating the capacity of the congested link. If there are any flows remaining over the link capacity, these flows are assigned to the next available path for the same origin, destination, PAT or PDT tuple in Step 3. The fully saturated link will be blocked by the infinite penalty (for the zero residual capacity) in Step 4. This algorithm continues until no congested link is left in the network. The solution of the algorithm will then be fed into the main transit assignment model as an initial feasible solution. To understand the proposed heuristic approach, we will show the algorithm on an example network in Figure 5.7.

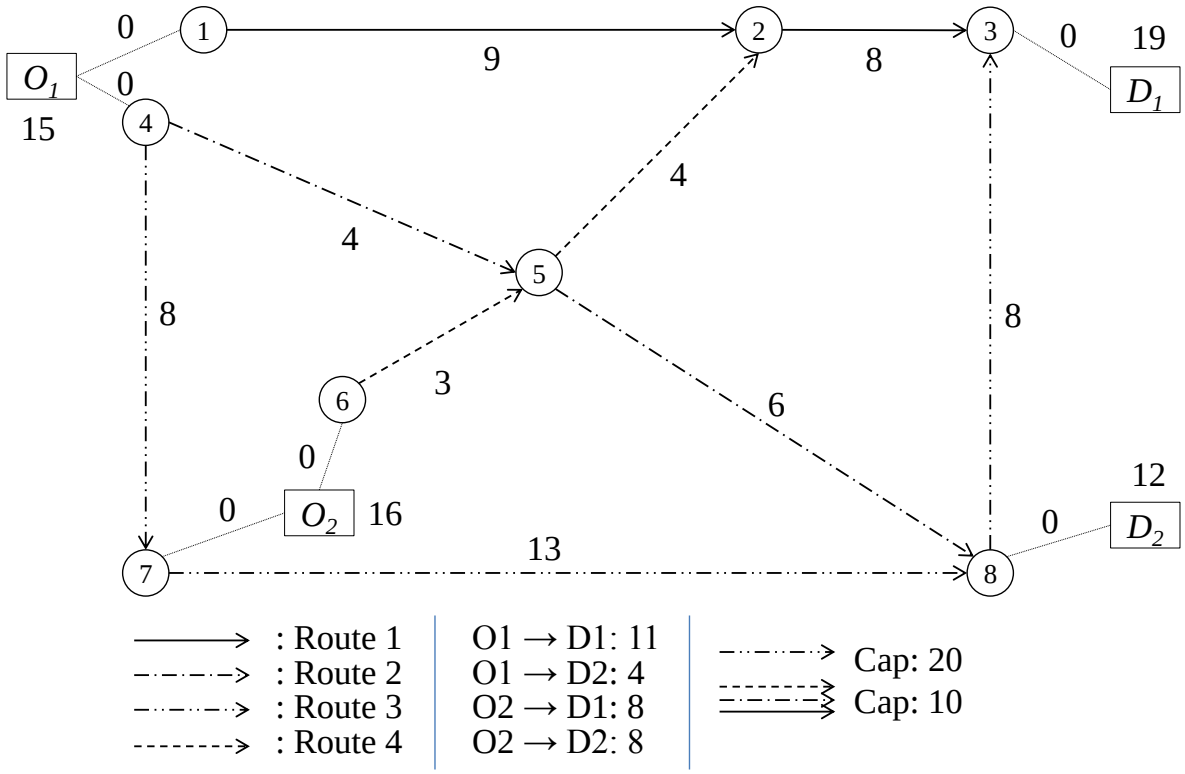
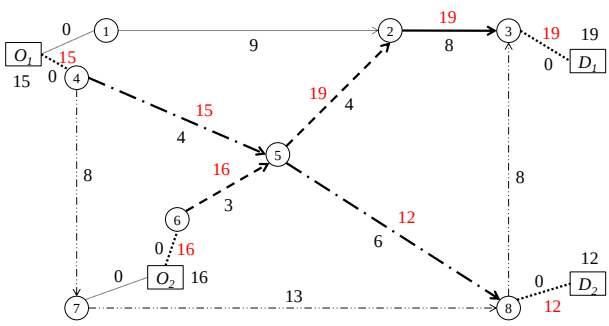
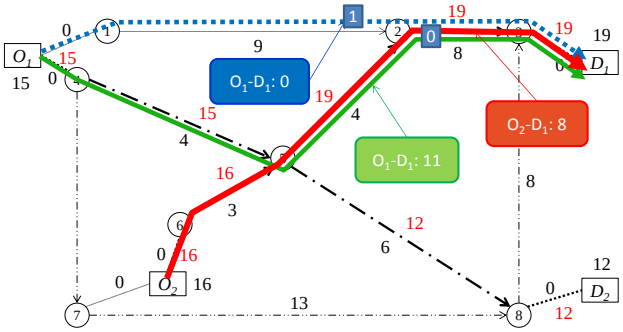


Figure 5.7 Example Network for a Better Initial Solution

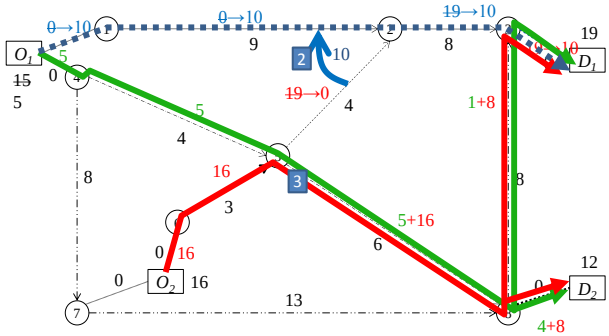
The network has two origins and two destinations which are connected by four transit routes, and the origin-destination demand and predefined capacity are shown on the bottom of the network in Figure 5.7. The numbers on each link represent link costs and the numbers on the origins (O_1 and O_2) and the destinations (D_1 and D_2) are the total origin and destination demand, respectively. Using the proposed heuristic, Figure 5.8 shows how to get a better initial solution considering the capacity of link and the priority of boarding according to each step of the algorithm.



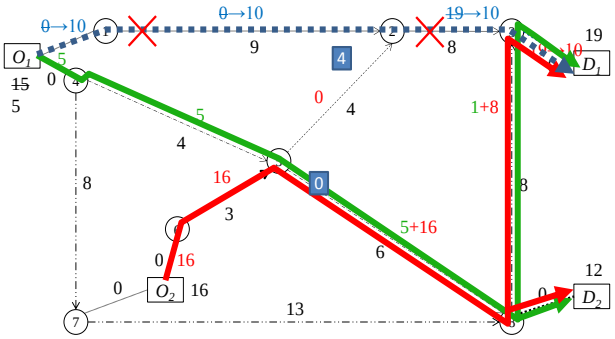
(a) Initial Loading and Congested Links



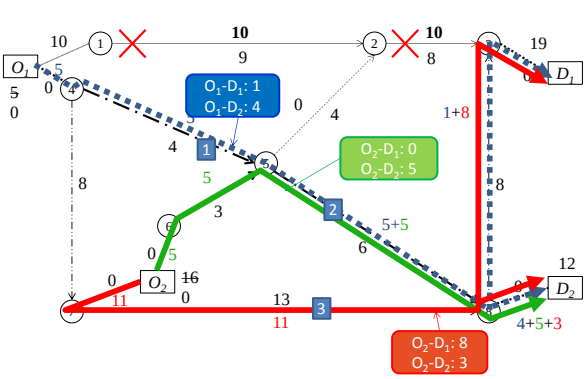
(b) Steps 0 and 1



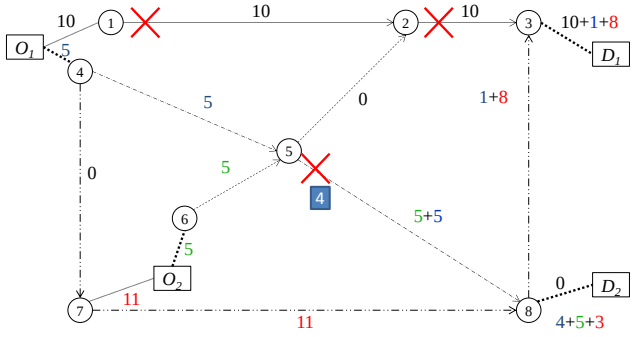
(c) Steps 2 and 3



(d) Steps 4 and 0 (again)



(e) Steps 1, 2, and 3 (again)



(f) Step 4 and Solution

Figure 5.8 Procedure of the Proposed Algorithm on A Example Network

The algorithm starts with an all-or-nothing assignment along the shortest paths for each O-D pair. In Figure 5.8(a), bold lines represent the congested links from this initial loading, and the numbers in red (other than travel times) are the initial flows from this all-or-nothing assignment. In Figure 5.8(b), selecting the first congested link closest to D_1, link (2-3), the algorithm searches if there is any priority path among the paths on link (2-3) coming from link (1-2). For this Step 1, predefined sets of paths for link (2-3) and link (1-2) are employed in which these path sets are prepared in advance for each link, by O-D pair and time interval by preferred arrival time (PAT) or preferred departure time (PDT). Using a given set of paths along the priority link (1-2), the dotted-line path for O_1 to D_1 will have 10 units of flow which are transferred from the path (O_1,1,2,3,D_1), saturating the capacity of link (2-3) in Step 2. This is shown in Figure 5.8(c). The overflowon this path is loaded on their next available paths that do not use link (2-3). In Step 3 of Figure 5.8(c), the remaining five units of flow from O_1 to D_1 are assigned on path (O_1,4,5,8,3,D_1) and the 16 units of flow leaving O_2 are assigned on the next available path (O_2,6,5,8,3,D_1), and on path (O_2,6,5,8,D_2). In Figure 5.8(d), the saturated links are deactivated by the infinite penalty cost for zero residual capacity according to Step 4. Then the proposed algorithm starts again searching for a congested link along Steps 0 to 4 shown in Figure 5.8 (d), (e), and (f). The number on each link in Figure 5.8(f) shows the feasible flows processed by the proposed bush-type heuristic. As commented earlier, the objective of the proposed algorithm is mainly for finding a better initial solution to expedite the solution procedure.

5.2.5.3 Stochastic Better Initial Solution (S-BIS)

Unlike the deterministic BIS, finding a better initial solution in a stochastic model is generally more difficult because it is harder to estimate how passengers will react to the capacity on congested links in the SUE solution. When we consider a stochastic better initial solution, we may think of two possible ideas: one is to utilize the deterministic user equilibrium (DUE) solution with D-BIS, and the other is to estimate the penalty directly according to the capacity on congested links.

The first idea, to utilize the existing D-BIS solution, focusses on sending enough flow to reach capacity on a congested link. However, the flows on the priority path on a capacitated link will be dissipated a bit by the stochastic effect on the path cost, through the entropy term on the path cost. However, if the entropy cost is not large enough to shift flows along a path alternative, the proposed D-BIS model may provide a similar solution to the deterministic UE. More detail of this model will be discussed in Chapter 7.4.3 regarding the gradient projection methodology.

For the second idea, we assume that the flows on each congested link do not exceed the capacity, as shown in Equation (5.7) and applied to the logit model below.

$$\sum_i p_i' = \frac{cap}{q_{rs}} \tag{5.7}$$

$$\sum_i \frac{\exp(u_i + k_i)}{\sum_j \exp(u_j + k_j)} = \frac{cap}{q_{rs}} \tag{5.8}$$

$$\begin{aligned} \sum_i \exp(u_i + k_i) &= \frac{cap}{q_{rs}} \sum_j \exp(u_j + k_j) \\ &= \frac{cap}{q_{rs}} \left[\sum_{j \neq i} \exp(u_j + k_j) + \sum_i \exp(u_i + k_i) \right] \end{aligned} \tag{5.9}$$

$$\left(1 - \frac{cap}{q_{rs}}\right) \sum_i \exp(u_i + k_i) = \frac{cap}{q_{rs}} \sum_{j \neq i} \exp(u_j) \tag{5.10}$$

Where, p': new path fraction with new penalty cost; i: paths feeding the capacitated link; k_i: new penalty cost for path i; u_i: utility of path i; q_{rs}: total demand from origin r to destination s; and cap: capacity of the congested link.

Let $w \equiv \frac{cap}{q_{rs}}$.

$$\sum_i \exp(u_i + k_i) = \frac{w}{(1-w)} \sum_{j \neq i} \exp(u_j) \tag{5.11}$$

If we say $k = k_i$ for all capacitated paths i, this means that every capacitated path has the same penalty cost regarding the capacity *cap*. Also, the capacity penalty k is derived through Equation (5.12) through (5.14).

$$\exp(k) \cdot \sum_i \exp(u_i) = \frac{w}{(1-w)} \sum_{j \neq i} \exp(u_j) \tag{5.12}$$

$$\exp(k) = \frac{w}{(1-w)} \cdot \frac{\sum_{j \neq i} \exp(u_j)}{\sum_i \exp(u_i)} = \frac{w}{(1-w)} \cdot \frac{\sum_{j \neq i} \exp(u_j) \Big/ \sum_{all} \exp(u)}{\sum_i \exp(u_i) \Big/ \sum_{all} \exp(u)} \tag{5.13}$$

Let $v \equiv \frac{\sum_i f_i}{q_{rs}}$ where f_i is the flow on path i, feeding into the congested link.

$$\therefore k = \ln\left[\frac{w}{(1-w)} \cdot \frac{(1-v)}{v} \right] \tag{5.14}$$

Applying this approach, we need to focus on the penalty cost k_i in terms of its asymmetric relation to the priority of boarding. If we have more alternative paths, the same penalties are assigned. When we solve the problem, the unknown asymmetric penalties can be estimated only if other predefined flows on a priority path are known. However, the flows on the priority path are only estimated with other given penalties according to the logit-model characteristics. In other words, the proportion of each path in a

logit model is estimated when all costs the paths are given. Also, another challenging task is to establish useful penalty cost models. This closed form of penalty estimation is not compatible with the proposed models using a hyperpath-based MSA or with path-based models since the models have a specific functions with parameters described in Chapter 5.2.4.

6 HYPERPATH-BASED ASSIGNMENT ON A TRANSIT SCHEDULE NETWORK

One of the possible transit assignment models utilizesthe proposed hyperpath model, considering a passenger's stochastic behavior. When we consider the hyperpath-based transit assignment, the main concern is on its solution technique with a logit-type choice model. We are interested in applying the method of successive averages (MSA) approach, since the MSA was implemented by Fisk (1981) in a logit-type stochastic auto assignment model and by Hamdouch and Lawphonpanich (2008) in a hyperpath (strategy)-based transit assignment on a transit schedule network. In addition, no appropriate solution algorithm accompanying this logit-type hyperpath has been introduced in the transit assignment other than MSA.

6.1 SUE Assignment Model Using a Logit-Based Hyperpath

With the assumed transit passenger behaviors and the proposed diagonalization, the transit assignment problem can be solved by a type of method of successive averages (MSA). For the first transit assignment problem, a hyperpath model is applied on a LBTE transit schedule network. Different from the hyperpath model used in the frequency-based approach which typically represents a passenger's "strategy" at each stop, we utilize a hyperpath as a path generation model. When we consider a weight function in a hyperpath (Gallo et al. 1993), we consider a type of log-sum function. Since the weighting function of a hyperpath is estimated by a log-sum model, the assignment model is classified as a stochastic user equilibrium (SUE) assignment when we employ the endogenous logit probability for flow assignment on the hyperpath. For the deterministic user equilibrium, the application of the hyperpath model is limited since the known schedule information does not create more than one alternative.

With this background, we will discuss the SUE model using the hyperpath on a LBTE transit schedule network. To consider the SUE model on a transit schedule network, we revisit the existing SUE model on an auto network. Fisk (1980) introduced the SUE problem by incorporating an entropy term. To apply this fundamental SUE model to a transit schedule network, we change the problem in terms of a link-to-link scheme. The problem can be simply extended to a link-based formulation incorporating a capacity constraint as shown in Equation (6.1) through (6.3).

$$Z = \sum_a \int_0^u c_a(w)dw + \sum_{ab} \int_0^u c_{ab}(w)dw + \frac{1}{\theta} \sum_{rs} \sum_p f_p(\ln f_p - 1) \tag{6.1}$$

$$\text{s.t. } \sum_{p \in P^{rs}} f_p^{rs} = D^{rs} \ \ \forall rs \in RS \tag{6.2}$$

$$f_p^{rs} \geq 0 \ \ \forall p \in P^{rs}, \forall rs \in RS \tag{6.3}$$

Where, $\sum_{ab} \int_0^{f_{ab}} c_{ab}(w)dw = \sum_{ab} c_{ab}^{trsf} f_{ab} + \sum_{ab} c_{ab}^{wait} f_{ab} + \sum_{ab} \int_0^{f_{ab}} c_{ab}^{cap}(w)dw$. Also, the transfer cost c_{ab}^{trsf} and waiting cost c_{ab}^{wait} are independent of the assigned flows and the capacity penalty cost is dependent on the flows. The model is solved with by a Lagrangian, $L = \sum_a \int_0^u c_a(w)dw + \sum_{ab} \int_0^u c_{ab}(w)dw + \frac{1}{\theta} \sum_{rs} \sum_p f_p(\ln f_p - 1) + \delta \sum_{rs} \sum_p (D^{rs} - \sum_p f_p^{rs})$. The necessary conditions to solve the Lagrangian are $\frac{\partial L}{\partial f_p} \geq 0$ and $f_p \frac{\partial L}{\partial f_p} = 0$. Since $f_p > 0$, $\frac{\partial L}{\partial f_p} = 0$, which gives the solution, $\delta^{rs} = c_p^{rs} + \frac{1}{\theta} \ln f_p^{rs}$. Also, the proportion of demand on path *p* gives the logit model, $\frac{f_p^{rs}}{D^{rs}} = \frac{e^{-\theta(c_p^{rs} - \delta^{rs})}}{\sum_{p' \in P^{rs}} e^{-\theta(c_{p'}^{rs} - \delta^{rs})}} = \frac{e^{-\theta c_p^{rs}}}{\sum_{p' \in P^{rs}} e^{-\theta c_{p'}^{rs}}}$.

When we consider the hyperpath model, the log-sum model generates a nest for every group of alternatives. With this hyperpath model, if we use the logit model for the flow dispersion on the hyperpath, this is equivalent to the logit probability being the conditional probability for each nest. The probability of path *p*, $\Pr_p$ using the hyperpath model is calculated using Equation (6.4) to (6.7).

$$\Pr_p = \prod_{k \in \bar{H}^{rs}} \left(\overline{\Pr}_k \cdot \delta_{kp} \right) \tag{6.4}$$

$$= \left(\frac{\bar{f}_{k_1}}{\bar{f}_r} \delta_{k_1 p} \right) \cdot \left(\frac{\bar{f}_{k_2}}{\bar{f}_{k_1}} \delta_{k_2 p} \right) \cdot \left(\frac{\bar{f}_{k_3}}{\bar{f}_{k_2}} \delta_{k_3 p} \right) \cdots \left(\frac{\bar{f}_s}{\bar{f}_{k_n}} \delta_{sp} \right) \tag{6.5}$$

$$= \frac{\bar{f}_s}{\bar{f}_r} \cdot \delta_{k_1 p} \cdot \delta_{k_2 p} \cdot \delta_{k_3 p} \cdot \cdots \cdot \delta_{sp} \tag{6.6}$$

$$= \frac{f_p}{D^{rs}} \tag{6.7}$$

The path probability is calculated from the probability of each nest or every link *k*, $\overline{\Pr}_k$, along hyperpath $\bar{H}^{rs}$ for origin-destination pair *rs*. The indicator δ_{kp} shows incidence between a nest link *k* and a path *p*, which is an elementary path in hypergraph $\bar{H}^{rs}$. Each probability of a nest is shown in $\bar{f}_{k_i} / \bar{f}_k$, $k_i \in \bar{F}_k^+$ for the forward link update, where $\bar{f}_k$ is the flow in nest *k*. Therefore, the outcome of the hyperpath for origin-destination pair *rs* is equivalent to the logit model. Also, the final solution satisfies the capacity constraint with priority, as well as following the logit probability with the capacity penalty.

The nest flow of origin link r in Equation (6.6), $\bar{f}_r$, is the demand D^{rs} in Equation (6.7), and the flow of destination nest link $\bar{f}_s$ is the flow of path p, f_p, as connected from the first alternative k_1 of origin link r to the destination link s, as indicated by the incidence values $\delta_{k_1 p} \cdot \delta_{k_2 p} \cdot \delta_{k_3 p} \cdot \cdots \cdot \delta_{sp}$.

To solve the capacitated SUE problem on a transit schedule network, we introduce a diagonalized MSA algorithm as shown in Figure 6.1.

Step 1: (Initialization)
- Find the least cost path
- Load flows on this path
- n=0 *

Step 2: (Capacity update)
- If sub-loop (from *Step 3 or Step 1*), then capacities are fixed and $n = n+1$.
- Else (from *Step 4*), residual capacities are changed by new flows and $n = 0$.

Step 3: (Diagonalization)
- Update the cost of network

Step 3.1: (Auxiliary Flows)
- Find the least cost hyperpath
- Load flows on the resulting hyperpath, creating (auxiliary flows)$^{n+1}$

Step 3.2: (Flows update: MSA)
- flows^{n+1} = 1 / (1+n)·(auxiliary flows)$^{n+1}$ + n / (1+n)·flows^{n}

Step 3.3: (Convergence Test)
- If satisfied, then go to Step 4.
- Else go to Step 1.

Step 4: (Convergence Test)
- If satisfied, then Stop.
- Else go to Step 1.

* n: iteration number

Figure 6.1 The Proposed MSA Algorithm Using Diagonalization

After the initial loading on a hyperpath in Step 1, the residual capacity is updated in Step 2, *but only* when the flows are converged in the inner loop, Step 3. Otherwise, the residual capacity of each link is *not* changed. Step 3 is the typical MSA-based flow updating process. Given the residual capacity from Step 2, auxiliary flows are created using the updated costs in the network, and new flows are generated by the auxiliary flows created iteratively in Steps 3.1 and 3.2. If the SUE convergence test in Step 3.3 is satisfied, the convergence of the outer loop (Step 4) is tested, comparing the difference in flows between the previous and the current iteration. As shown by Sheffi (1984), this guarantees a SUE flow pattern.

6.2 Transit Passenger Behavior in a Logit-Type Model

6.2.1 Overlap on a Transit Schedule Network

When we define a unit of overlap in an auto network, it is often using a unit of distance or perhaps travel time. In a multimodal transportation network, Hoogendoorn-Lanser et al. (2005) explored meaningful overlapping units such as time, distance, and, "trip leg" or simply "leg", which is typically accompanied by a transfer between two different modes such as intercity (IC) train, express train, tram, and bus. Among the overlapping units, they showed the advantage of using "leg". However, the definition of overlap in a transit schedule network should be different, especially for an intra-urban trip. The overlap in a transit schedule network happens only with a transit run (or trip) made by an individual transit vehicle, typically with a transfer. This overlapping length can be measured by physical distance or time of each leg or of a number of legs as used in Hoogendoorn-Lanser et al. (2005). Alternately, the leg can be specified as a "time-dependent leg" accompanying a transfer and a preferred arrival time (PAT) or departure time (PDT). Also, Hoogendoorn-Lanser et al. (2007) utilized the separate trip leg for the inter-urban trip which is separated into an access mode (normally with walk and bike), an egress mode with local bus and tram, and a main transit mode by intercity or express train. In an urban area, it is hard to

identify a primary mode, between bus, heavy rail, light rail, etc. In the following sub-chapter, we will talk about the available route choice model for this problem with overlaps.

6.2.2 Route Choice Models Considering Overlap

When the logit model is used in route choice, the overlapping of routes produces a violation of the Independence of Irrelevant Alternatives (IIA) property. This problem in route choice has been studied by many researchers, including Cascetta et al. (1996), Vovsha (1997), Ben-Akiva and Bierlaire (1999), Prashker and Bekhor (1998, 2000), Hoogendoorn-Lanser et al. (2005, 2007), and Prato (2009). Among the logit route choice models that consider corrections for this overlapping, such as C-Logit, path-size logit (PSL), cross-nested logit, and generalized extreme value (GEV) models, we chose the PSL model, based on the success of Hoogendoorn-Lanser et al. (2005, 2007).

6.2.2.1 Path-Size Logit (PSL) Model

This PSL model was introduced by Ben-Akiva and Bierlaire (1999) as shown in Equation (6.8) to (6.10). According to Prato (2009), the path size of a path *p* is defined to be the magnitude of the overlapping length divided by the full path length, or (l_a / l_p), but including all overlapping links as indicated in Equation (6.9). From Hoogendoorn-Lanser et al. (2005, 2007), it is possible to penalize an alternative path with longer length more heavily, using $\left(l_p / l_j\right)^{\gamma}$ in Equation (6.10), where γ and β are parameters to be calibrated for the model.

$$P_p = \frac{\exp(C_p + \beta \cdot \ln PS_p)}{\sum_{j \in Q} \exp(C_j + \beta \cdot \ln PS_j)} \tag{6.8}$$

$$PS_p = \sum_{a \in A_p} \frac{l_a}{l_p} \cdot \frac{1}{\sum_{j \in Q} \delta_{aj}} \tag{6.9}$$

$$PS_p = \sum_{a \in A_p} \frac{l_a}{l_p} \cdot \frac{1}{\sum_{j \in Q} \left(\frac{l_p}{l_j} \right)^{\gamma} \delta_{aj}} \tag{6.10}$$

According to Equations (6.9) and (6.10), the path size correction must satisfy $PS_p \leq 1.0$, in which PS_p = 1.0 means a unique path without overlapping, and Equation (6.8) resolves to a simple multinomial logit model (Prato, 2009). This PSL model was used in a multimodal transportation network by Hoogendoorn-Lanser et al. (2005, 2007). In terms of applying the PSL model on this study, path-size can be simply added to the objective function in Equation (6.1). This result in the problem defined in Equation (6.11) to (6.14). In Equation (6.12), the penalty cost is derived from the function Equation (5.3).

$$\min\ Z = \sum_a \int_0^u c_a(w)dw + \sum_{ab} \int_0^u c_{ab}(w)dw + \frac{1}{\theta} \sum_{rs} \sum_p f_p (\ln f_p - 1) + \beta \sum_{rs} \sum_p f_p \ln PS_p \tag{6.11}$$

$$= \sum_a c_a f_a + \sum_{ab} \frac{e^{-\alpha \cdot r_b}}{\alpha \cdot r_b} \left(\left(f_{ab} - \frac{1}{\alpha} \right) e^{\alpha \cdot f_{ab}} + \frac{1}{\alpha} \right) + \frac{1}{\theta} \sum_{rs} \sum_p f_p (\ln f_p - 1) + \beta \sum_{rs} \sum_p f_p \ln PS_p \tag{6.12}$$

$$\text{s.t.} \sum_{p \in P^{rs}} f_p^{rs} = D^{rs} \quad \forall rs \in RS \tag{6.13}$$

$$f_p^{rs} \geq 0 \quad \forall p \in P^{rs}, \forall rs \in RS \tag{6.14}$$

As before, we can utilize a Lagrangian to solve the problem, and the solution for the SUE flows will be:

$$\lambda^{rs} = c_p + \frac{1}{\theta} \ln f_p + \beta \ln PS_p \tag{6.15}$$

Where, $c_p = \sum_a c_a \delta_{ap} + \sum_{ab} c_{ab} \delta_{abp}$. The Lagrange multiplier of the problem captures the entropy term of the optimal path and the additional overlapping correction. The resulting fraction of flows in the solution satisfies the logit model with PSL:

$$\frac{f_p}{D^{rs}} = \frac{e^{-\theta\left(c_p - \lambda^{rs} + \beta \ln PS_p\right)}}{\sum_q e^{-\theta\left(c_q - \lambda^{rs} + \beta \ln PS_q\right)}} = \frac{e^{-\theta\left(c_p + \beta \ln PS_p\right)}}{\sum_q e^{-\theta\left(c_q + \beta \ln PS_q\right)}} \tag{6.16}$$

We need an additional process to estimate the "length" in the path size correction for overlapped paths. According to Hoogendoorn-Lanser et al. (2005), on a multimodal system, the length used in the PSL model is dependent on the transportation access mode-leg (walking or biking), main transit mode (intercity or express train), and mode-leg for egress (tram, Metro, or bus), for an inter-urban multimodal system. However, in intra-urban travel, the transit travel pattern will be different from the inter-urban area since (a) access and egress time is relatively smaller than that for inter-urban trips, (b) a mode-specific leg may not apply easily because the same mode may have multiple runs (or trips) for a time-dependent condition. Yet, we may consider that each leg is defined by a transfer. For this reason, we apply a transfer-based leg for the PSL length. When we consider each transfer leg, we define each leg using the hierarchical path search mentioned earlier (Figure 4.12). [We note that calibration of the PSL model is not considered in this study.]

6.3 Application on an Example Network

6.3.1 Example

For a test, first we utilize a simple network as shown in Figure 6.2 (the same as in Figure 5.7). We assume that there are four routes running over 11 transit stops (numbered from 1 to 11) between two origins (O_1

and O_2) and two destinations (D_1 and D_2). All links between two consecutive stops are transit links except for four transfer links (9,2), (5,10), (10,5), and (11,8). The other links connecting from origins or to destinations are access and egress links. For the schedule of transit service on this network, we simply assume one schedule per each route which is shown at each stop. These times are assumed to be either scheduled arrival or scheduled departure times. Numbers over links show the walking times of access, egress, and transfer. Demand is shown below the network. Total origin demands are 15 and 16 for origins O_1 and O_2, respectively. Total destination demands are 19 and 12 for destination D_1 and D_2, respectively. Capacity is assumed to be 10 units of flow for Routes 1, 2, and 3, and Route 4 has capacity of 20.

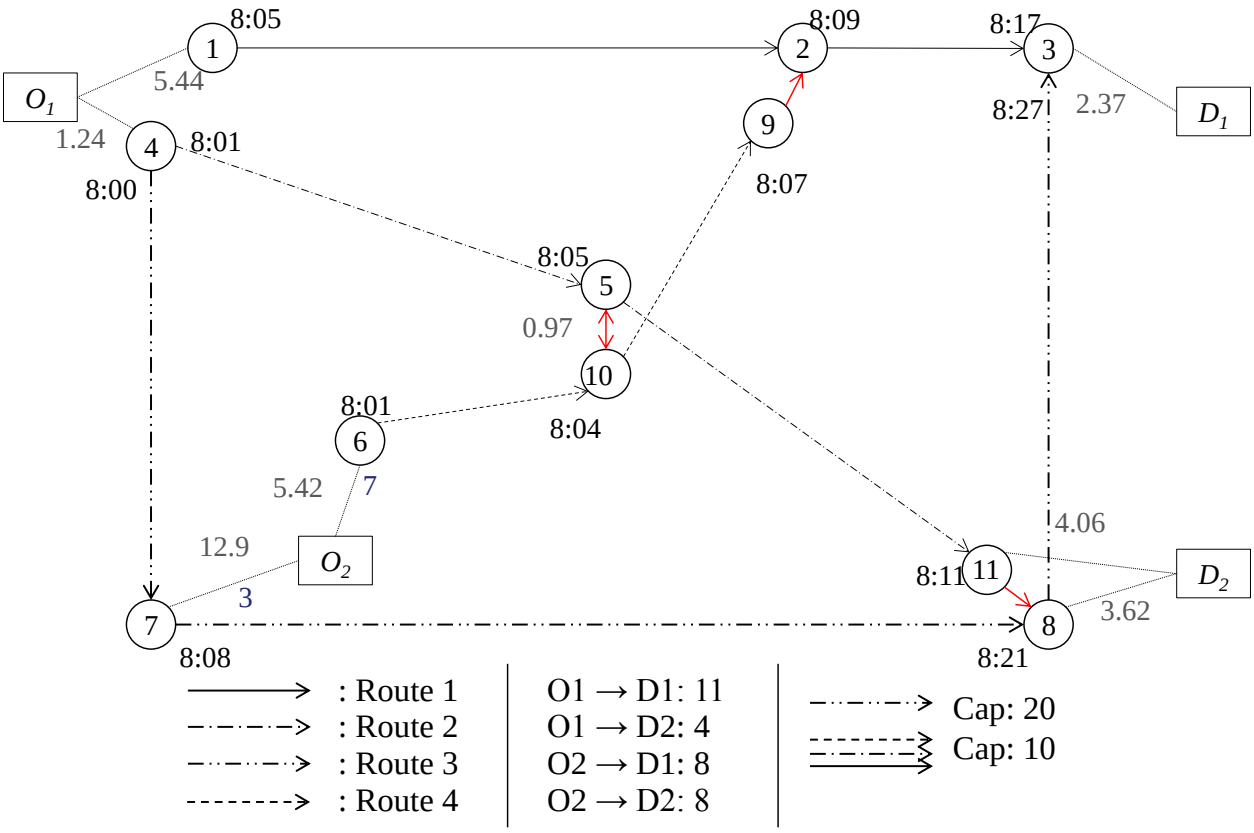
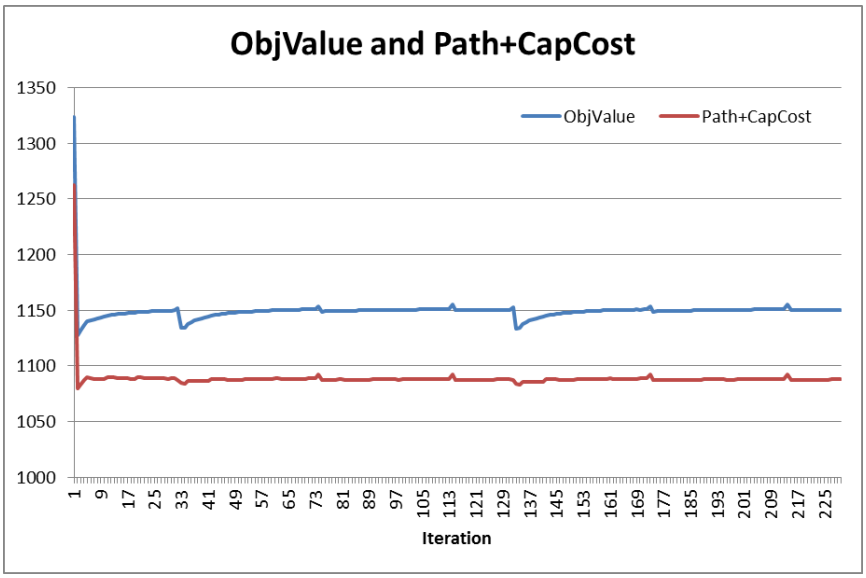
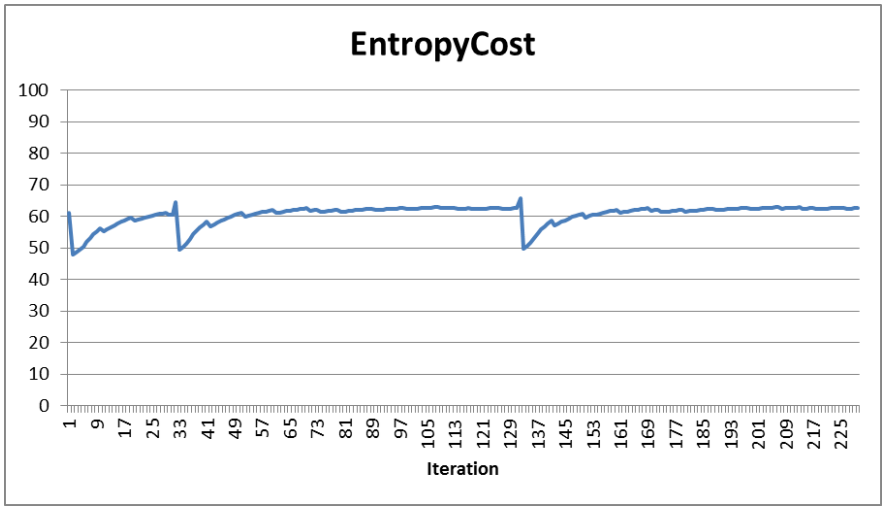


Figure 6.2 Simple Network for Test

Recall that the Hyperpath-based MSA assignment model is fundamentally utilized for stochastic passenger behavior. The objective function value and its path-only and capacity cost are shown in Figure 6.3(a). “Path+CapCost” represents the sum of the path-only cost and the capacity-cost as in the first and second term of Equation (6.1), or alternately as the objective cost excluding the entropy cost, which is the third term of Equation (6.1). One noticeable pattern in the hyperpath-based model is its relatively stationary convergence rate by iterations after a huge drop in the first iteration. This is mainly due to searching a new hyperpath according to the capacity cost. At every iteration after the first drop, a new hyperpath is searched, according to the capacity cost from the previous iteration, which does not allow sharp increases to the capacity cost in the objective function. Through the outer iterations, “ObjValue” shows several drops. Figure 6.3(b) shows the entropy cost around each diagonalization iteration. After completing a diagonalization iteration (the first outer iteration), the MSA method has a major role in distributing the flows regarding the loading results of the previous diagonalization, to decrease entropy costs.



(a) Objective Value and Path+Capacity Cost



(b) Entropy Cost

Figure 6.3 Objective and Other Costs of Hyperpath-based Assignment Model

More comparison results and analyses for this example network is continued in Chapter 10.2.

7 PATH-BASED ASSIGNMENT ON A TRANSIT SCHEDULE NETWORK

As a major alternative to the hyperpath-based transit assignment, we consider a path-based model for a passenger's stochastic behavior as well as for a passenger's deterministic behavior. The main reason to consider a path-based solution method, especially for the logit-type choice model, is that the path-based approach has two main advantages. First, it is easy to incorporate the stochastic component, called the entropy cost. We also note that this stochastic model can be directly converted to a deterministic model by eliminating the entropy term. Second, the path enumeration increases the performance of the model since the shortest path search is relatively simple, and the size of set is relatively small when using a compatible PAT or PDT.

This study also introduces the better initial solution (BIS) mainly for improving the shortcomings of diagonalization in this path-based assignment model. To consider the asymmetric cost relation, we propose a diagonalization technique introduced by Sheffi (1985). The critical procedure of diagonalization is updating the residual capacity according to the previous UE or SUE solution, so that the method requires outer iterations for diagonalization, called a diagonalization iteration, and inner iterations for DUE or SUE in a diagonalization iteration as utilized in the hyperpath-based assignment model. As the diagonalization steps iterate, the residual capacity of every link is updated and converges as the assigned link flows of two successive diagonalization iterations get closer to each other. However, this assignment model requires relatively longer diagonalization iterations, especially when the initial flows are assigned on non-priority paths, since all the flows in the non-priority paths should be transferred onto the priority paths. This inefficient approach can be improved by searching an initial solution by loading flows along the priority paths over congested links. We call this model of finding a good initial solution, the "better initial solution" (BIS), model which is introduced in 5.2.5 and considered for both deterministic and stochastic assignment, deterministic BIS (D-BIS) and stochastic BIS (S-BIS).

We will talk more about the proposed path-based assignment model in the following chapters.

7.1 Path Cost with Capacity Penalty

If we assume that the residual capacity is fixed, the cost function, including transfer time and waiting time, is as formulated in Equations (7.1) and (7.2), yielding a basic schedule-based transit assignment. The capacity cost penalty creates a "soft capacity" constraint since the capacity constraint is relaxed to allow flows that exceed the vehicle capacity, and a capacity penalty function c_{ab}^{cap} is included in the objective function (7.1).

$$\min\ Z = \sum_a \int_0^u c_a(w)dw + \sum_{ab} \int_0^u \left(c_{ab}^{trsf}(w) + c_{ab}^{wait}(w) + c_{ab}^{cap}(w)\right)dw \tag{7.1}$$

$$\text{s.t.} \sum_p f_p = D^{rs} \quad \forall rs \tag{7.2}$$

Since the link cost, transfer, and waiting times (or costs) are constant, we can modify the objective function in Equation (7.1) by explicitly representing the capacity penalty from Equation (5.3). This yields Equation (7.3).

$$\min\ Z = \sum_a c_a\ f_a + \sum_{ab} c_{ab}^{trsf} f_{ab} + \sum_{ab} c_{ab}^{wait} f_{ab} + \sum_{ab} \frac{e^{-\alpha \cdot r_{ab}(\mathbf{f})}}{\alpha \cdot r_{ab}(\mathbf{f})} \left(\left(f_{ab} - \frac{1}{\alpha} \right) e^{\alpha \cdot f_{ab}} + \frac{1}{\alpha} \right) \tag{7.3}$$

In Equation (7.3), the link cost is c_a, transfer cost is c_{ab}^{trsf}, waiting cost is c_{ab}^{wait} and $r_{ab}(\mathbf{f})$ stands for the residual capacity of link *ab*, and α is a parameter for the capacity cost. When we solve the problem as a Lagrangian, the Lagrange multiplier can be derived as in Equation (7.4), which means that UE can be determined using the path cost. Transfer, waiting, and capacity (penalty) costs are aggregated into c_{ab} for simplification of the link-to-link cost in Equation (7.5).

$$\lambda^{rs} = c_p = \sum_a c_a \delta_{ap} + \sum_{ab} c_{ab} \delta_{abp} \tag{7.4}$$

$$c_{ab} = c_{ab}^{cap} + c_{ab}^{trsf} + c_{ab}^{wait} \tag{7.5}$$

Using the Lagrange multiplier, each path is searched using link cost c_a and turning penalty cost c_{ab}, and this makes our proposed schedule-based assignment model compatible with a path-based assignment approach.

7.2 Gradient Projection for a Path-Based Assignment

In the UE problem, instead of using a classical link-based assignment solution like the Frank-Wolfe algorithm, Jayakrishnan et al. (1999) proposed a path-based assignment model which restates Beckman's UE objective and constraints in terms of non-negative, non-shortest paths, according to the Goldstein-Levitin-Poljak gradient projection by Bertsekas (1976, 1982). This takes the form of a quasi-Newton approximation, as shown in Equation (7.6) to (7.8).

$$\min \tilde{Z}(\tilde{f}) = \sum_a \int_0^{\tilde{f}} c_a(w)\, dw \tag{7.6}$$

$$\text{s.t.} \quad f_{\tilde{p}} \geq 0 \;\; \forall f_{\tilde{p}} \in \tilde{F} \tag{7.7}$$

$$\left(f_{\bar{p}} = D^{rs} - \sum_{\tilde{p} \neq \bar{p}} f_{\tilde{p}} \quad \forall rs \in RS \right) \tag{7.8}$$

The UE model of Beckman (1952) is represented in terms of a vector of non-shortest path flows, $\widetilde{F}$ with each non-shortest path flow represented by $f_{\tilde{p}}$. The shortest path flow $f_{\bar{p}}$ represents the demand for each origin-destination pair *rs* after excluding the non-shortest path flows.

This path-based approach was also utilized in the SUE problem. Bekhor and Toledo (2005) introduced a stochastic version of the auto assignment model. Fundamentally, an entropy term (Chen, 1999) is applied, and Bekhor and Toledo (2005) solve the path-based SUE model based on Jayakrishnan's approach by *a priori* paths enumerated by a *k*-shortest path method.

7.3 Deterministic User Equilibrium (DUE) Model

To solve a transit path-based assignment with capacity constraints and passenger priority, we introduce a deterministic schedule-based transit assignment that can be solved using the gradient projection method, based on the model formulation shown in Equation (7.9) to (7.12).

$$\min \ \widetilde{Z} = \sum_{a} \int_0^f c_a(w)dw + \sum_{ab} \int_0^f c_{ab}(w)dw \tag{7.9}$$

$$= \sum_{a} c_a \ f_a + \sum_{ab} c_{ab}^{trsf} f_{ab} + \sum_{ab} c_{ab}^{wait} f_{ab} + \sum_{ab} \frac{e^{-\alpha \cdot r_{ab}}}{\alpha \cdot r_{ab}} \left(\left(f_{ab} - \frac{1}{\alpha} \right) e^{\alpha \cdot f_{ab}} + \frac{1}{\alpha} \right) \tag{7.10}$$

$$\text{s.t.} \quad f_{\tilde{p}} \geq 0 \ \ \forall f_{\tilde{p}} \in \widetilde{F} \tag{7.11}$$

$$\left(f_{\bar{p}} = D^{rs} - \sum_{\tilde{p} \neq \bar{p}} f_{\tilde{p}} \quad \forall rs \in RS \right) \tag{7.12}$$

The idea is to apply a quasi-Newton method, meaning that the projection will be structured mainly with the first and second derivatives of the objective function. When we assume path sets P_{rs} and $P_{\overline{\overline{rs}}}$ according to origin-destination pair rs and $\overline{\overline{rs}}$, and paths $\tilde{p}, \bar{p} \in P_{rs}$ and $\tilde{q}, \bar{q} \in P_{\overline{\overline{rs}}}$, if origin-destination pair $rs = \overline{\overline{rs}}$, path incidence values $\delta_{\tilde{p}\tilde{q}} = 1$ or 0 depending on whether the paths are identical, and $\delta_{\bar{p}\bar{q}} = 1$, $\delta_{\bar{p}\tilde{q}} = 0$, and $\delta_{\tilde{p}\bar{q}} = 0$, since there is a single shortest path. But if $rs \neq \overline{\overline{rs}}$, by similar arguments, $\delta_{\tilde{p}\tilde{q}} = 0$, $\delta_{\bar{p}\bar{q}} = 0$, $\delta_{\bar{p}\tilde{q}} = 0$, and $\delta_{\tilde{p}\bar{q}} = 0$. Considering this property, in the gradient projection method, the first and second derivatives are restated in terms of non-shortest-path flows, as in Equations (7.13) through (7.16).

$$g_{\tilde{p}} \equiv \frac{\partial \tilde{Z}}{\partial f_{\tilde{p}}} = \frac{\partial Z}{\partial f_p} - \frac{\partial Z}{\partial f_{\bar{p}}} = \sum_a c_a (\delta_{a\tilde{p}} - \delta_{a\bar{p}}) + \sum_{ab} c_{ab} (\delta_{ab\tilde{p}} - \delta_{ab\bar{p}}) = c_{\tilde{p}} - c_{\bar{p}} \tag{7.13}$$

$$\bar{h}_{\tilde{p}\tilde{q}} \equiv \frac{\partial^2 \tilde{Z}}{\partial f_{\tilde{p}} \partial f_{\tilde{q}}} = \sum_a \frac{\partial c_a}{\partial f_a} (\delta_{a\tilde{p}} - \delta_{a\bar{p}})(\delta_{a\tilde{q}} - \delta_{a\bar{q}}) + \sum_{ab} \frac{\partial c_{ab}}{\partial f_{ab}} (\delta_{ab\tilde{p}} - \delta_{ab\bar{p}})(\delta_{ab\tilde{q}} - \delta_{ab\bar{q}}) \tag{7.14}$$

$$h_{\tilde{p}\tilde{q}} \equiv \frac{\partial^2 \tilde{Z}}{\partial f_{\tilde{p}}^{\,2}} = \sum_a \frac{\partial c_a}{\partial f_a} (\delta_{a\tilde{p}} - \delta_{a\bar{p}})^2 + \sum_{ab} \frac{\partial c_{ab}}{\partial f_{ab}} (\delta_{ab\tilde{p}} - \delta_{ab\bar{p}})^2 \tag{7.15}$$

$$= \sum_{ab} c'_{ab} (\delta_{ab\tilde{p}} - \delta_{ab\bar{p}})^2 = \sum_{ab \in \bar{p} \cup \tilde{p}} c'_{ab} \tag{7.16}$$

Where, $c'_{ab} \equiv \dfrac{\partial c_{ab}}{\partial f_p} = \sum_{ab \in p} \dfrac{\partial c_{ab}}{\partial f_{ab}} \delta_{abp} = \sum_{ab \in p} \dfrac{\partial \left(\dfrac{f_{ab}}{r_{ab}} \exp\left(\alpha\left(f_{ab} - r_{ab}\right)\right) \right)}{\partial f_{ab}} \delta_{abp}$. The values of δ in Equations (7.12) to (7.16) involve the incidence of a particular in-vehicle or transfer link on a given path.

As mentioned earlier, gradient projection in the path-based model can be applied by using an approximation of the Hessian matrix. In this path-based assignment model, we can consider using a full matrix or just the diagonal elements of the Hessian matrix, given in Equations (7.14) and (7.15), respectively. The diagonal elements of the Hessian matrix can be simplified using Equation (7.16) just summing the first derivative of capacity cost on the shortest path $\bar{p}$ and non-shortest path $\tilde{p}$ except for overlapped link *ab*.

Iteratively, flows are updated by the gradient projection as shown in Equations (7.17) and (7.18) by a full Hessian or diagonal elements of the Hessian matrix to find a solution. As mentioned in Jayakrishnan et al. (1994), the flows that are calculated to be negative will be projected to zero, thereby satisfying the non-negativity constraint.

$$f_{\tilde{p}}^{n+1} = \max\left[0,\ f_{\tilde{p}}^{n} - \alpha\left(\frac{\partial^2 \tilde{Z}}{\partial {f_{\tilde{p}}^{n}}^2}\right)^{-1} \cdot (d_{\tilde{p}}^{n} - d_{\bar{p}})\right] = \max\left[0,\ f_{\tilde{p}}^{n} - \alpha \cdot g_{\tilde{p}}^{n} / \bar{h}_{\tilde{p}\tilde{q}}^{n}\right] \tag{7.17}$$

$$f_{\tilde{p}}^{n+1} = \max\left[0,\ f_{\tilde{p}}^{n} - \alpha \cdot g_{\tilde{p}}^{n} / h_{\tilde{p}\tilde{q}}^{n}\right] \tag{7.18}$$

According to the fundamental characteristic that shortest path flows are determined by the non-shortest path flows, it is worth noting that the Hessian matrix consists only of non-shortest paths. Also, it is worth noting that a zero Hessian could occur if there is no capacity violation.

To consider path-based assignment models using a full Hessian matrix and diagonal elements associated with the vehicle capacity constraint, we propose a full Hessian and a diagonalized gradient projection algorithm as shown in Figure 7.1. The main difference from the hyperpath-based MSA algorithm lies in preparing a set of paths for each O-D pair using the hyperpath search model. This is because the hyperpath model provides appropriate alternative paths according to the headways of each route, and this

predefined set of paths improves performance in considering relatively long outer iterations. Using a full Hessian matrix is reasonable since the algorithm utilizes relatively simple steps as shown in Figure 7.1(a). Steps 0 through 4 show the typical process of transportation assignment except for Step 2, which uses a full Hessian matrix of the non-shortest path costs.

Using the hyperpath in Figure 7.1(b) also improves the performance of algorithm. The gradient projection model using diagonalization starts with an all-or-nothing assignment in Step 0. Step 1 adjusts the residual capacities on the links in the network. However, the residual capacities are only updated after the flows in the previous iteration reach a diagonalized UE (for a single O-D pair), which is represented in Step 2 (Diagonalization). After a convergence check in Step 2.3, the algorithm will determine if the upper level (outer loop) convergence exists in Step 3; i.e., whether there is a UE for all O-D pairs. If not, the residual capacity of each link will be updated by the new flows. These iterations continue until a deterministic UE is accomplished over all O-D pairs.

Step 0 (Initialization)
- Search an initial set of alternative paths for each O-D pair
- Search the least cost path and load flows on the searched path

Step 1 (Direction)
- Search the least cost path

Step 2 (Move)
- Update new non-shortest path flows using a full Hessian of non-shortest path costs for each O-D

Step 3 (Convergence Test)
- If satisfied, then Stop;
- Else then go to *Step 1*

(a) Using a Full Hessian Matrix

Step 0 (Initialization)
- Search an initial set of alternative paths for each O-D pair
- Search the least cost path and load flows on the searched path

Step 1 (Capacity updates)
- If sub-loop (from *Step 2 or Step 1*), then do not update residual capacities
- Else (from *Step 3*), then residual capacities are changed by new flows

Step 2 (Diagonalization)
- Update the link costs and turn penalties in the network

Step 2.1 (Direction) Search the least cost path

Step 2.2 (Move) Update new flows

Step 2.3 (Convergence Test)
- If satisfied, then go to *Step 3*;
- Else then go to *Step 1*

Step 3 (Convergence Test)
- If satisfied, then Stop;
- Else then go to *Step 1*

(b) Using Diagonal Elements with a Diagonalization Technique

Figure 7.1 Solution Algorithms Using Gradient Projection

For a side note for the full Hessian path-based assignment, we introduce some background. By exploring the asymmetric cost relation, we proposed a path-based assignment model using a full Hessian scaling matrix. We also introduce a diagonalization technique by updating the residual capacity at each new diagonalization iteration, according to the UE or SUE solution of the previous iteration. However, we realize that the diagonalization technique may have relatively longer iterations, although path-based assignment guarantees the advantages on its performance. In terms of replacing the proposed diagonalization technique, we propose a full Hessian scaling matrix with the path-based assignment model.

7.4 Stochastic User Equilibrium (SUE) Model

7.4.1 "Sinkhole" Effect

With the capacity constraint, we propose a SUE assignment and solution methodology using a method similar to the deterministic gradient projection. The SUE objective function and Lagrange multipliers for a schedule-based transit assignment are as follows.

$$Z = \sum_a \int_0^u c_a(w)dw + \sum_{ab} \int_0^u c_{ab}(w)dw + \frac{1}{\theta} \sum_{rs} \sum_p f_p(\ln f_p - 1) \tag{7.19}$$

$$\hat{c}_p^{rs} = c_p^{rs} + \ln(f_p^{rs}) \tag{7.20}$$

In terms of a path-based model, the objective function captures the path cost with an entropy term for the assigned flows on path *p*. Chen (1999) and Bekhor and Toledo (2005) introduced this entropy-type cost, derived from the aggregate SUE approach developed by Fisk (1980). However, these SUE models have difficulties with the entropy term. Specifically, the flows on a path can be small enough to be effectively

zero. Unfortunately, these small flows make the cost of a path go to negative infinity, and all the flows are assigned to that path. In the model application, this zero flow is temporarily assumed to be *one* unit, which also allows a temporary change in flow by gradient projection. By achieving positive flows, the gradient projection model continues in the normal process. To do this, the original objective function is also modified to add *one* to the flow in the entropy term, as shown in Equation (7.21).

$$Z = \sum_a \int_0^u c_a(w)dw + \sum_{ab} \int_0^u c_{ab}(w)dw + \frac{1}{\theta}\sum_{rs}\left(\sum_{p|f_p \neq 0} (f_p)\left[\ln(f_p) - 1\right] + \sum_{p'|f_{p'}=0} (f_{p'}+1)\left[\ln(f_{p'}+1) - 1\right]\right) \quad (7.21)$$

The consideration of such a "sinkhole" where flows are zero also affects to the gradient and Hessian matrix as shown in Equations (7.22) and (7.23).

$$\frac{\partial \tilde{Z}}{\partial f_{\tilde{p}}} = \sum_a c_a(\delta_{a\tilde{p}} - \delta_{a\bar{p}}) + \sum_{ab} c_{ab}(\delta_{ab\tilde{p}} - \delta_{ab\bar{p}}) + \frac{1}{\theta}[\ln(f_{\tilde{p}}+1) - \ln(f_{\bar{p}})] \quad (7.22)$$

$$\frac{\partial^2 \tilde{Z}}{\partial f_{\tilde{p}} \partial f_{\tilde{q}}} = \sum_a \frac{\partial c_a}{\partial f_a}(\delta_{a\tilde{p}} - \delta_{a\bar{p}})(\delta_{a\tilde{q}} - \delta_{a\bar{q}}) + \sum_{ab} \frac{\partial c_{ab}}{\partial f_{ab}}(\delta_{ab\tilde{p}} - \delta_{ab\bar{p}})(\delta_{ab\tilde{q}} - \delta_{ab\bar{q}})$$
$$+ \frac{1}{\theta}\left(\frac{1}{f_{\tilde{p}}+1}\delta_{\tilde{p}\tilde{q}} + \frac{1}{f_{\bar{p}}}\delta_{\bar{p}\bar{q}} \right) \quad (7.23)$$

The diagonal approximation of the Hessian is given in Equation (7.24).

$$\frac{\partial^2 \tilde{Z}}{\partial f_{\tilde{p}}^{\,2}} = \sum_a \frac{\partial c_a}{\partial f_a}(\delta_{a\tilde{p}} - \delta_{a\bar{p}})^2 + \sum_{ab} \frac{\partial c_{ab}}{\partial f_{ab}}(\delta_{ab\tilde{p}} - \delta_{ab\bar{p}})^2 + \frac{1}{\theta}\left(\frac{1}{f_{\tilde{p}}+1} + \frac{1}{f_{\bar{p}}} \right) \quad (7.24)$$

Where, the flows $f_{\tilde{p}}$ are assumed to have zero flow.

The "sinkhole" effect is also projected in the first and second derivatives. The flows close to zero do not generate infinite values in the first and second derivatives, respectively, thereby satisfying the non-negative flow constraint.

7.4.2 Path-Size Logit (PSL) for Overlapping Problem

Like the hyperpath model, when we utilize the logit-type model, overlapping is an IIA violation to be resolved. Equation (7.25) includes the path-size logit (PSL) term as in Equation (6.11) and constraints are the same as in the hyperpath model.

$$Z = \sum_a \int_0^u c_a(w)dw + \sum_{ab} \int_0^u c_{ab}(w)dw + \frac{1}{\theta}\sum_{rs}\sum_p (f_p)\left[\ln(f_p) - 1\right] + \beta \sum_{rs}\sum_p f_p \ln PS_p \qquad (7.25)$$

In the assignment algorithm, the PSL is added to the enumerated paths after the path search on all O-D pairs. Since the paths are enumerated in advance, additional PSL modification in the middle of algorithm is not required.

7.4.3 Stochastic Better Initial Solution (S-BIS)

When we consider the diagonalized gradient projection model, the D-BIS (BIS on Deterministic UE) model is intuitively easy to apply since the assumption that passengers with priority will board until the transit vehicle reaches capacity. On the other hand, estimating a BIS for the SUE model is generally hard because we need to figure out the relation between stochastic passenger behaviors with priority and the vehicle capacity, as mentioned in Chapter 5.2.5.3.

For the estimation of S-BIS, let f_i^* and c_i^* be the optimal flow andthe resulting cost on path *i* in the DUE solution. Considering the entropy cost, the stochastic passenger assignment will have a cost

$c_i' = c_i^* + \ln(f_i^*)$ at the DUE solution. This shows the simple phenomenon that a path having greater flow will have higher cost with the entropy term, and other paths having lesser flows will have lower entropy cost, inducing lower path cost as well.

Instead of simply obtaining the solution of the DUE, it is also possible to utilize the D-BIS by assuming $\hat{f}_i = f_i^*$ and $c_i' = \hat{c}_i + \ln(\hat{f}_i)$, where $\hat{f}_i$ and $\hat{c}_i$ are the pseudo-flow and cost of path *i* at D-BIS, and c_i' is assumed to be zero when the flow $\hat{f}_i$ is zero. This phenomenon will create a flow shift, $f_i' = \hat{f}_i + \Delta f_i$. We are more interested in the flow shift Δf_i from the flows $\hat{f}_i$, especially on the congested link and priority paths. To build a compatible flow shift, we also utilize the gradient projection to estimate f_i', specifically $f_{\bar{p}}' = \max\left[0, \hat{f}_{\bar{p}} - \alpha \cdot g / h\right]$ where, $\bar{p}$ and $\tilde{p}$ are the shortest path and non-shortest path according to the cost c_i' for every path *i*, and *g* and *h* are the gradient and Hessian, respectively. The proposed algorithm to solve for the SUE-BIS is shown in Figure 7.2. For a better estimation, parameter α can be adjusted, although it is normally set to 1.0.

Step 0: Get the D-BIS solution using Figure 5.6

Step 1: Update the cost of path *i* by $c_i' = \max\left[0, \hat{c}_i + \ln(\hat{f}_i)\right]$.

Step 2: Search the min. cost and update the flow of each path *i* according to gradient projection

Step 3: Update the capacity of each path according to the flows and set the initial capacity for SUE model

Figure 7.2 Stochastic Better Initial Solution (S-BIS) Algorithm

However, this S-BIS does not guarantee a better initial solution since it is hard to estimate the residual capacity of each path in the SUE solution. It may be better to use D-BIS directly, instead of using S-BIS in some cases if the entropy cost is relatively small relative to the path cost c_p, or if demand is strongly disaggregate, with specific PAT or PDT and alternative paths are a relatively narrow set, with a significant path with reasonably high probability. In other words, these cases show that the SUE solution may be fairly close to the DUE solution.

7.5 Applications

7.5.1 Example

For the simple model test, we applied the same transit network as shown in Figure 6.2.

7.5.2 Deterministic Gradient Projection Model Results

7.5.2.1 Diagonalization Model

In the deterministic model using gradient projection, as shown in Figure 7.3, the first iteration normally has a huge increase in the objective value caused by an all-or-nothing assignment, which violates the vehicle capacity. The objective value then drops sharply within several iterations. In this initial huge increase and decrease in Figure 7.3, we notice that the capacity cost is the main contributor to the objective value in this solution method.

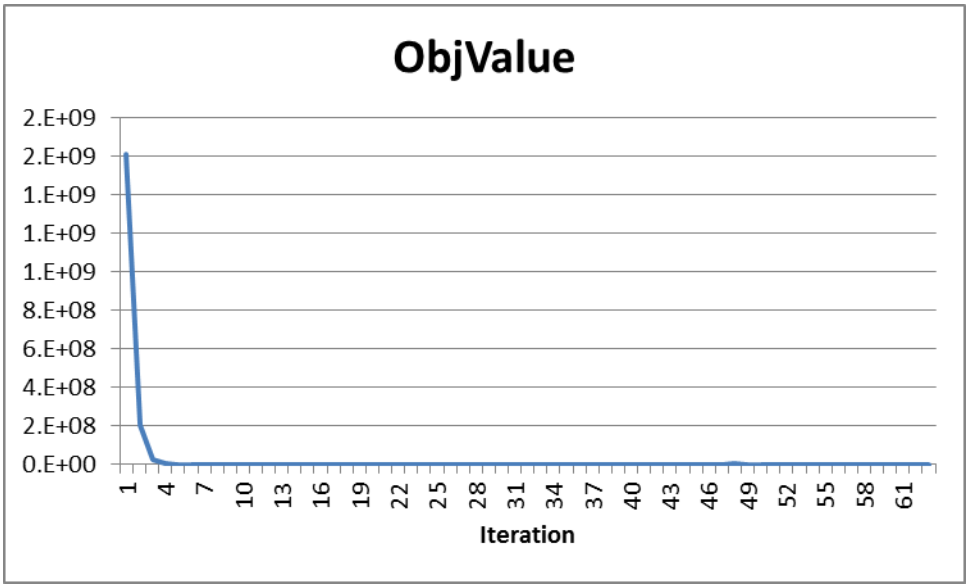


Figure 7.3 Objective Cost of Deterministic Gradient Projection Model

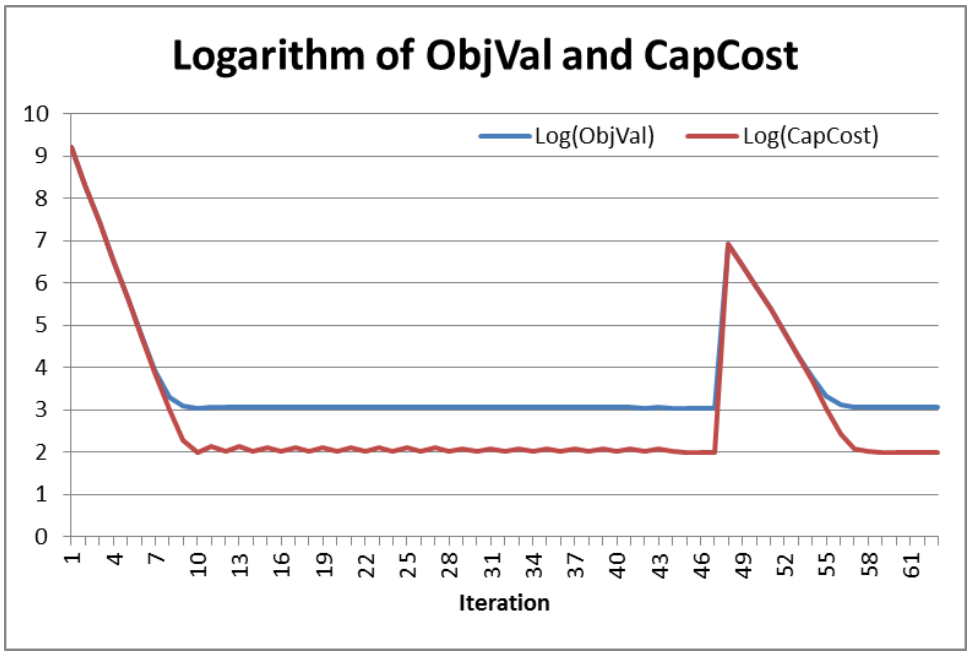


Figure 7.4 Objective Value and Capacity Cost of the Diagonalization Model

For the diagonalization model, Figure 7.4 also shows a strong bond between the objective function value and the capacity cost, normally with jumps occurring at the major diagonalization iterations. Related to two diagonalization iterations, Iteration 1 to 8 and 48 to 55, are evidently dominated by the capacity cost. The capacity cost shows a moderate oscillation along the iterations other than these two segments.

The other significant costs are the path cost (see Figure 7.5). The path cost shows a moderate increase and then a gradual decrease compared to the capacity cost. Again, we can notice two major leaps during iteration 1 ~ 8 and 48 ~ 55, similar to the leaps in the capacity cost portion in Figure 7.4. The main reason of the two leaps are related to the increasing use of detoured paths: the capacity cost along congested stops shifts flows to less congested alternative routes, which finally increase the actual total path cost while decreasing the capacity cost.

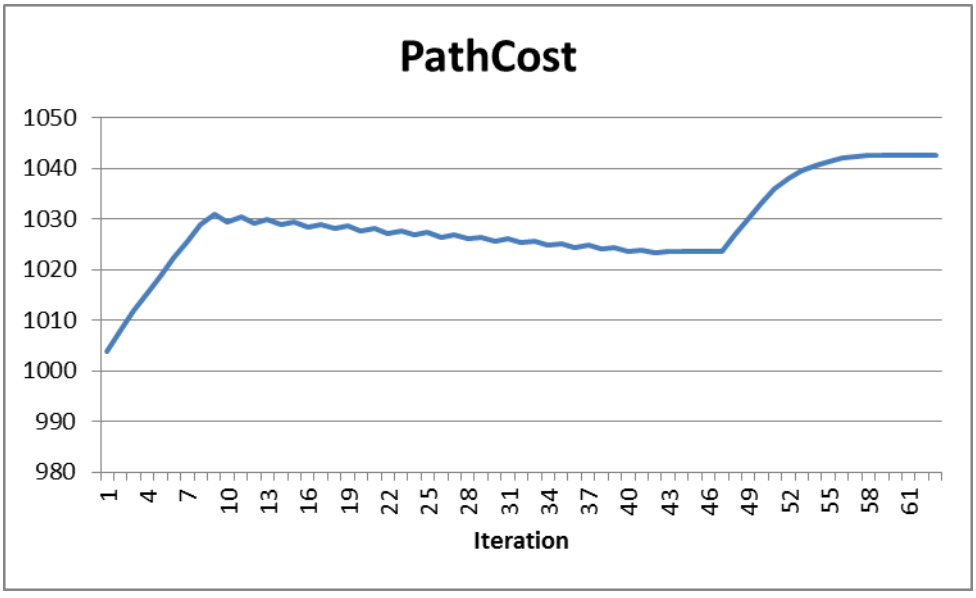


Figure 7.5 Path Cost in the Deterministic Objective Function

7.5.2.2 BIS (D-BIS) Diagonalization Model

For a better performing model, we discussed BIS models for deterministic and stochastic passenger behavior in Chapters 5.2.5 and 7.4.3. The application of the D-BIS model on the simple network shows the result in Figure 7.6. The key difference from Figure 7.4 is, after the first huge objective cost increase, only one sharp decrease is detected, according to the logarithmic representation, Considering the main benefit of utilizing the BIS model is to cut computation time, the objective cost change with one big increase and decease along iterations makes sense. As we can see in Figure 7.6, the capacity cost dominates iteration 1 through 26, then converges with the path cost of the objective function. Also Figure 7.7 shows the difference between deterministic non-BIS and BIS models, especially in decreasing the number of iterations.

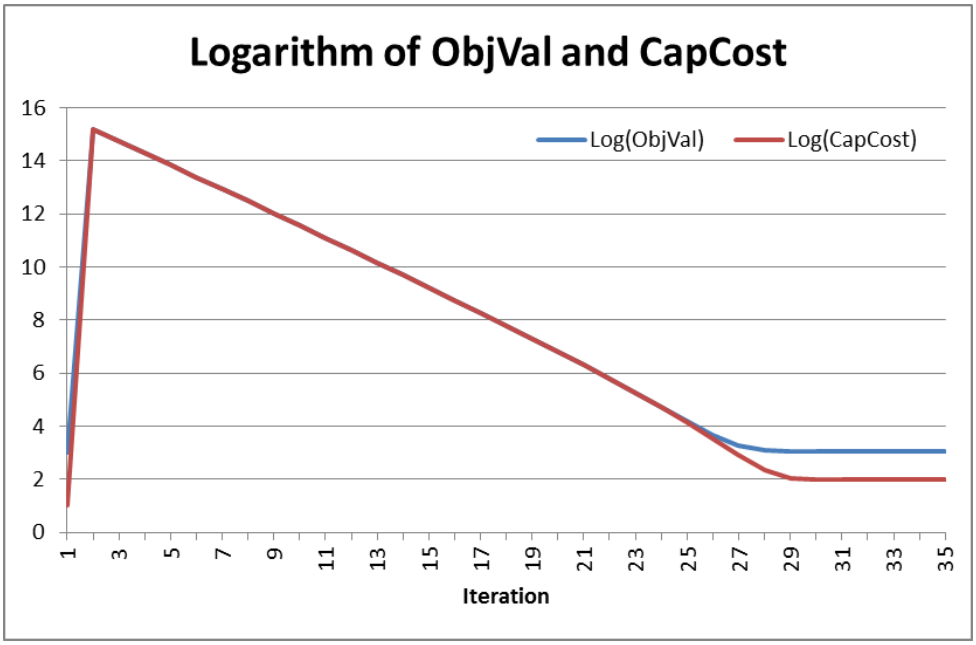


Figure 7.6 Logarithmic Objective and Capacity Cost of D-BIS model

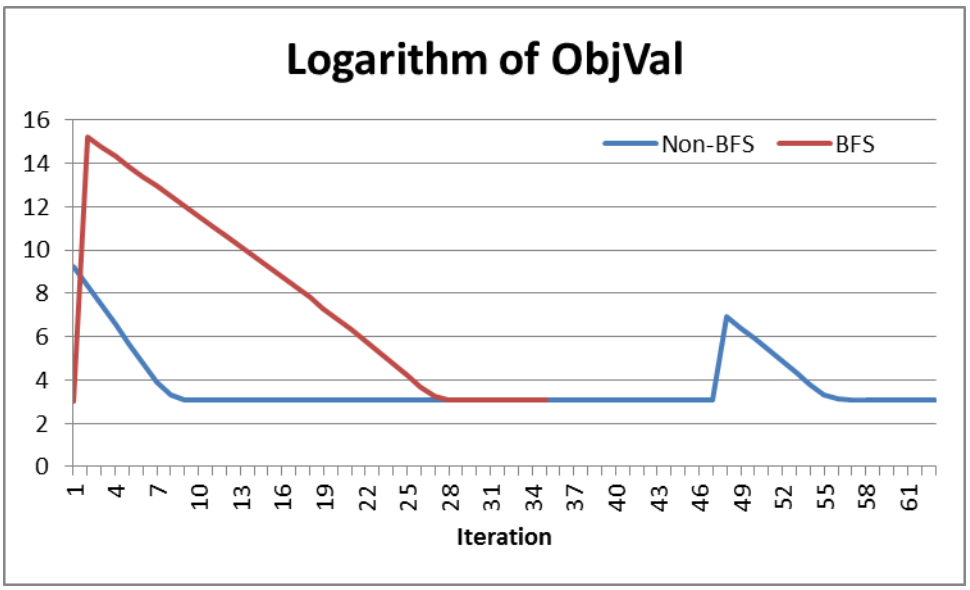


Figure 7.7 Logarithmic Objective Value of Non-BIS and BIS Model

7.5.2.3 Full Hessian Model

For another gradient projection method, we considered a full Hessian for the scaling matrix of the path-based assignment model. Considering the asymmetric cost relation directly without any diagonalization steps, this approach shows a relatively gradual decrease (compared with the diagonalization model) in the logarithmic objective cost, as shown in Figure 7.8. One small increase of the objective cost (or capacity cost) is detected at iteration 54, and compared with the diagonalization method, we realize that it takes more iterations to converge.

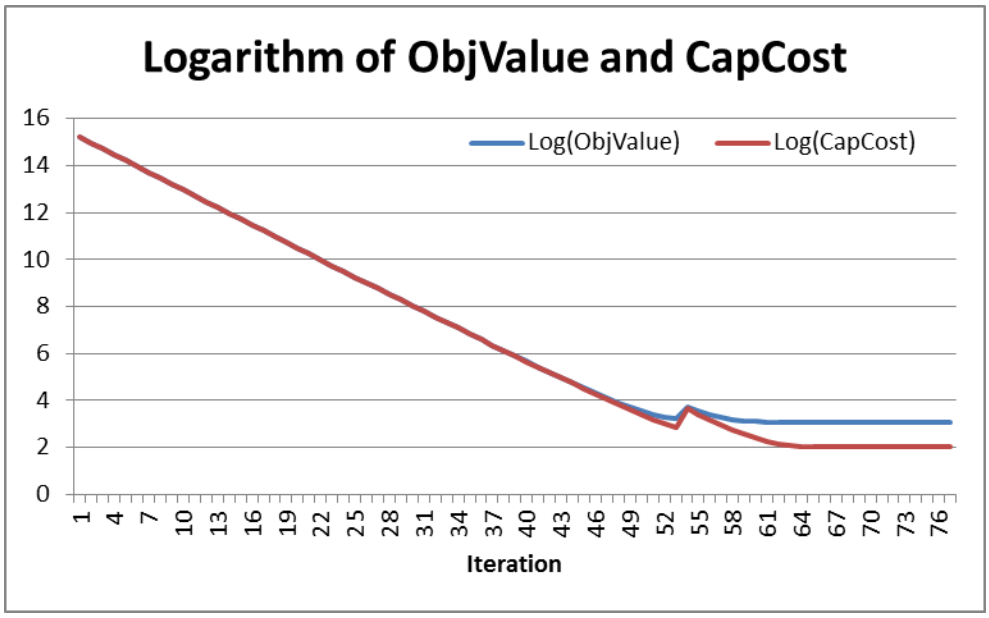


Figure 7.8 Logarithmic Objective Cost and Capacity Cost in Deterministic Full Hessian Model

The path cost in the objective function shows several distinct sudden decreases along a gradual increase over iterations as shown in Figure 7.9. When we consider that the path cost could include a huge capacity cost, these decreases are not evident in Figure 7.8 (except the final drop), with the logarithmic value of objective cost in total. The drops in the path cost originate from the fact that capacity cost changes affect passenger flows on longer paths more gradually, but affect shorter paths much more quickly. These flow changes momentarily drop the overall path costs over the network.

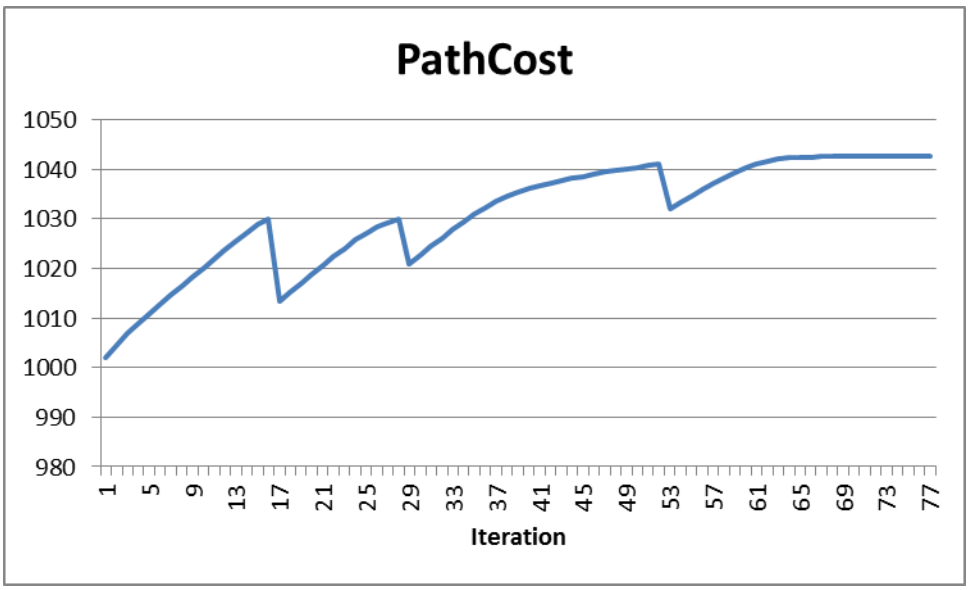


Figure 7.9 Path Cost in Objective Function of Deterministic Full-Hessian Model

7.5.3 Stochastic Model Results

7.5.3.1 Diagonalization Model

In the stochastic model, the diagonalization model shows an evident pattern of big leaps at each diagonalization iteration. As shown in Figure 7.10, two huge increases are observed during two major diagonalization iterations. After the first decrease, minor oscillations are also detected and we notice a small increase in the objective cost after the second peak between iterations 62 and 82. The same pattern which was observed in the deterministic diagonalization model (see Figure 7.4) is also detected in this stochastic model as shown in Figure 7.11, although it takes more iterations to converge.

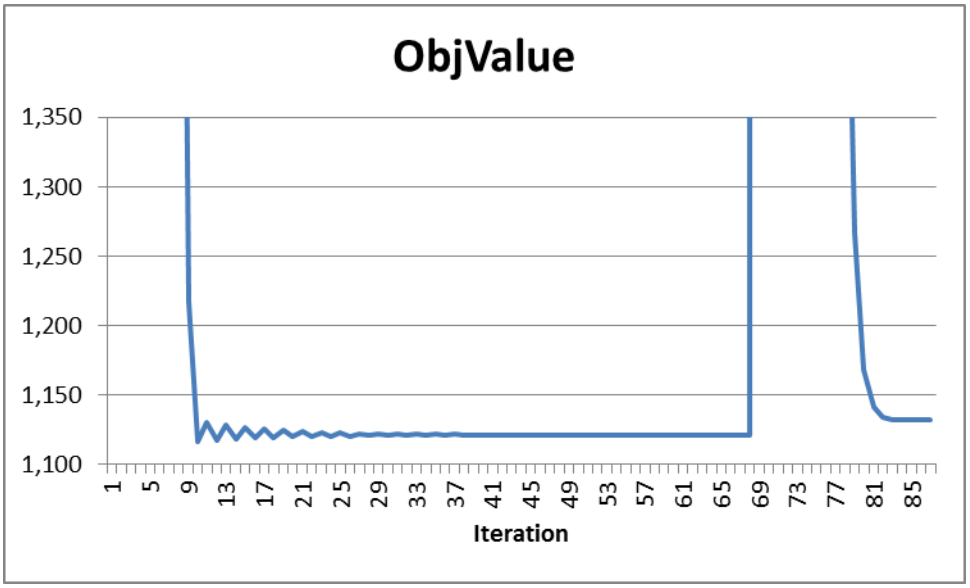


Figure 7.10 Objective Cost of Stochastic Gradient Projection Model Using Diagonalization

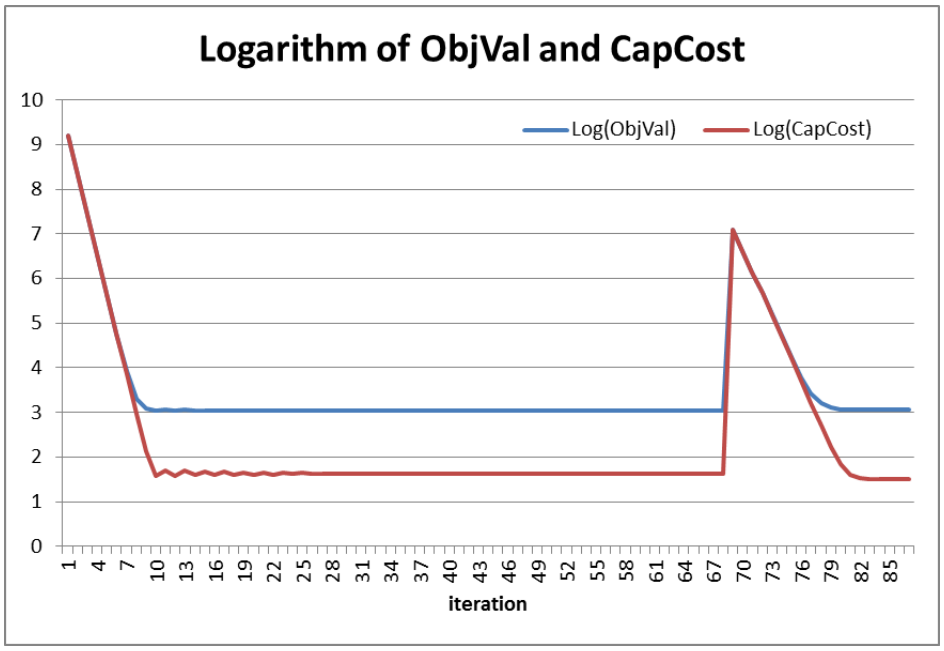


Figure 7.11 Logarithmic Objective and Capacity Cost of Stochastic Diagonalization Model

7.5.3.2 BIS Diagonalization Model

As we mentioned in Chapter 7.4.3, it is also worth applying D-BIS for the stochastic model, if the capacity cost is a major cost element and if the entropy cost is weak enough. For this reason, we applied D-BIS on the stochastic diagonalization model. As shown in Figure 7.12, after one increase, the proposed stochastic BIS model (actually D-BIS) decreases the objective cost as well as the capacity cost. Similar to the deterministic model, relatively short iterations are noticed in Figure 7.12.

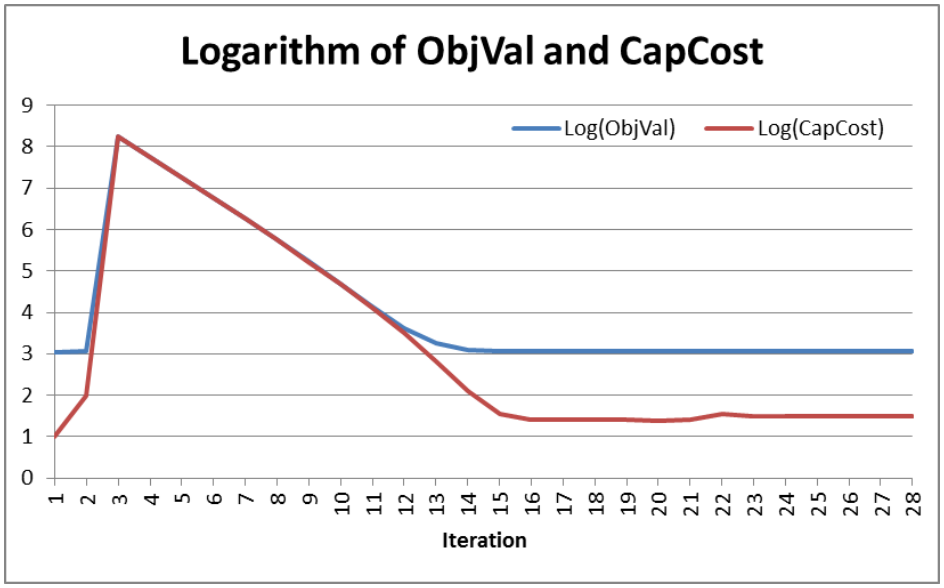


Figure 7.12 Logarithmic Objective and Capacity Cost of Stochastic BIS Model

Figure 7.13 also shows a similar comparison between non-BIS and BIS models to those shown in Figure 7.7. This pattern also shows the potential value to utilize D-BIS for the stochastic model.

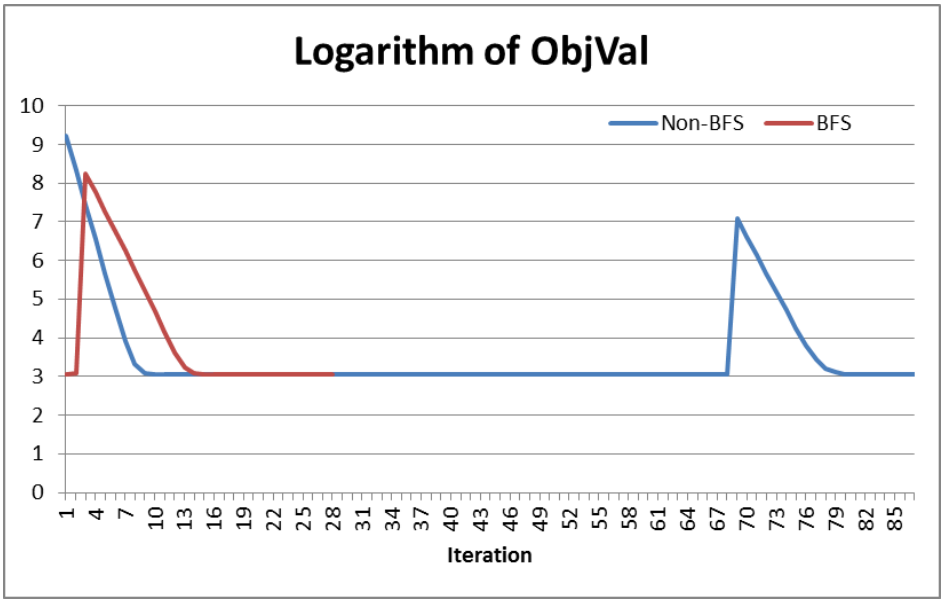


Figure 7.13 Logarithmic Non-BIS and BIS Objective Cost

7.5.3.3 Full Hessian Model

Different from the deterministic full Hessian method, the proposed stochastic model in Figure 7.14 shows a smoother decrease, but following the same pattern as the deterministic model.

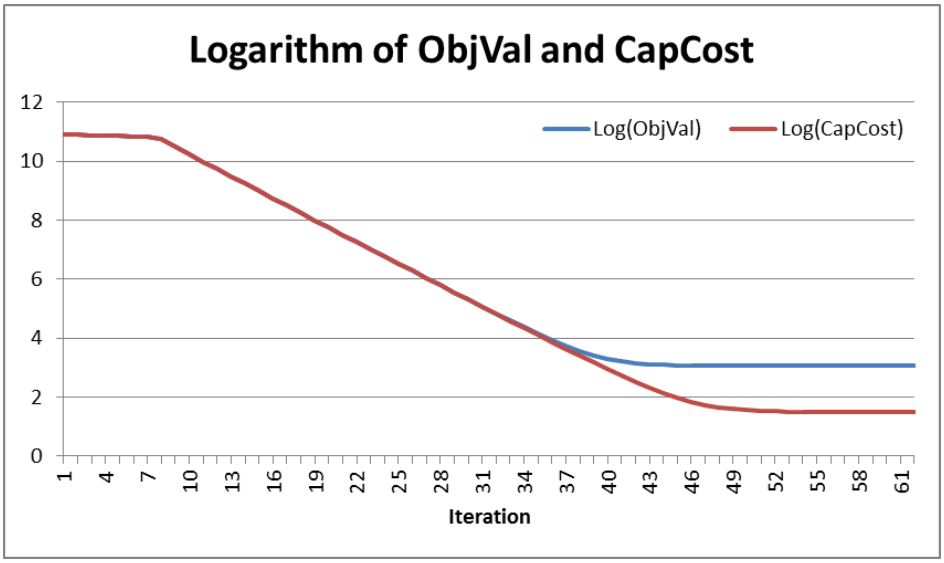


Figure 7.14 Logarithmic Objective Cost and Capacity Cost in Stochastic Full Hessian Model

The path cost shown in Figure 7.15 represents a smoother increase than the deterministic model, with one distinctive decrease which is not noticeable in the logarithmic objective and capacity cost in Figure 7.14. Again, this indicates that there are passengers' path changes dropping the path cost, according to changes in the capacity cost when passengers shift to alternative paths.

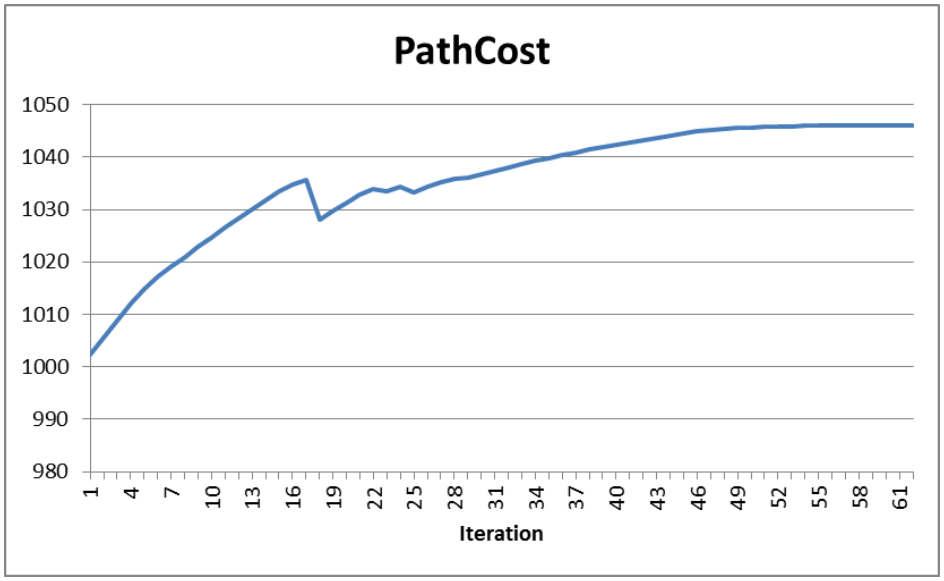


Figure 7.15 Path Cost in Objective Function of Stochastic Full-Hessian Model

More comparison results and analyses for this example network will be continued in Chapter 10.2.

8 TRANSIT ASSIGNMENT USING SELF-ADAPTIVE GRADIENT PROJECTION

We considered the hyperpath- and path-based assignment models in the previous chapters. In this chanpter, we explore one more model, a self-adaptive projection model, to replace the diagonalization technique and the full Hessian path-based method. We realized that we may face some slow performance with the fullHessian path-based model. The slow performance happens when the paths cross each other with multiple transfers, and the priority at each transfer point can be different, shifting flows from a non-priority path, but the shortest path, to the priority path during some iterations. For this reason, we introduce the self-adaptive projection model, which has a different approach based on the Lipschitz continuity condition and focuses on its step-size.

8.1 Self-Adaptive Gradient Projection

When we consider a Newton-type approximation method in transportation assignment, a representative case using a scaling matrix with the Hessian matrix in the gradient projection method (introduced in Chapter 7) is shown in Equation (8.1).

$$\mathbf{f}^{n+1} = \mathbf{f}^{n} - \alpha^{n} \nabla Z(\mathbf{f}^{n}) \cdot \mathbf{H}(\mathbf{f}^{n})^{-1} \tag{8.1}$$

Where, $\mathbf{f}^{n}$ stands for a vector of flows at iteration *n*, $\nabla Z(\mathbf{f}^{n})$ and $\mathbf{H}(\mathbf{f}^{n})$ are the gradient and Hessian matrix of vector $\mathbf{f}^{n}$, and α^{n} is a step-size parameter. In this approach, the parameter α^{n} is assumed to have a constant over the iterations *n*, say 1.0. On the other hand, we may consider a different approach to approximate the flows by changing the step-size, α^{n}. For this problem, He et al. (2002)

introduced a modified Goldstein–Levitin–Polyak (GLP) method based on Bertsekas (1976), and Chen et al. (2012) also proposed a modified GLP method. To understand the effect that this step-size parameter has on the algorithm convergence, we start with the Lipschitz continuity condition and the required step-size from Equation (8.2) to (8.4). The step-size α^n needs to critically satisfy Equation (8.2).

$$0 < \alpha_L^n \le \alpha^n \le \alpha_U^n < 2\frac{\kappa}{L^2} \tag{8.2}$$

Where, α_L^n and α_U^n are lower and upper bounds, L is the Lipschitz constant satisfying the Lipschitz continuity condition shown in Equation (8.3), and κ is the uniform modulus in which $\mathbf{c}(\mathbf{f})$ satisfies a one-side Lipschitz continuity condition shown in Equation (8.4) for global convergence.

$$\left\|\mathbf{c}(\mathbf{f}^n) - \mathbf{c}(\mathbf{f}^{n+1})\right\| \le L\left\|\mathbf{f}^n - \mathbf{f}^{n+1}\right\| \tag{8.3}$$

$$\left(\mathbf{f}^n - \mathbf{f}^{n+1}\right)^T \left(\mathbf{c}(\mathbf{f}^n) - \mathbf{c}(\mathbf{f}^{n+1})\right) \le \kappa\left\|\mathbf{f}^n - \mathbf{f}^{n+1}\right\|^2 \tag{8.4}$$

Considering the difficulty of estimating L and α^n, Bertsekas (1976) suggested a generalized Armijo's rule (Armijo, 1966) for the suitable step-size α^n. Given a non-stationary point $\mathbf{f}^n$, search $\alpha^n = \beta^{l^n} u$ satisfying Equation (8.5).

$$\frac{Z(\mathbf{f}^n) - Z(\mathbf{f}^n(\beta^{l^n} u))}{\mathbf{f}^n - \mathbf{f}^n(\beta^{l^n} u)} \ge \sigma \nabla Z(\mathbf{f}^n) \tag{8.5}$$

Where, $\beta \in (0,1)$, $u > 0$ and $\sigma \in (0,1)$ are fixed scalars, and l^n will be the smallest integer number for which the inequality of Equation (8.5) holds.

The modified GLP model by Chen et al. (2012) uses the gradient projection method, but also embeds the Lipschitz continuous conditions of Equations (8.3) and (8.4) in a modified GLP form. The proposed model by Chen et al. (2012) is called the self-adaptive gradient projection (SAGP) algorithm as shown in Figure 8.1. After the initial setup for the scalars and parameters in Step 0, Bertsekas' Armijo rule was modified in terms of non-shortest path flows, and the shortest path flows are simply updated by the difference between the total demand on the O-D pair and total non-shortest path flows updated by new step-size α^{n+1} in Step 1. Finding a new step-size α^{n+1} with the new flows will be continued until (Condition 1) is satisfied; that condition is derived from applying the gradient projectionwith the Lipschitz continuity conditions in Equations (8.3) and (8.4). "Condition 2" in Step 2 allows a non-monotonic sequence of the step-size, compared with the current step-size α^{n}. This process between Step 1 and 2 is continued until satisfying the convergence test in Step 3.

Step 0: Initialization

- Set $\delta \in (0,1)$, $u \in [0.5,1]$, $\varepsilon > 0$, $\alpha^{\max}, \alpha^0 > 0$, and any feasible flows $\mathbf{f}^0$
- Set $\gamma^0 = \alpha^0$ and $n = 0$

Step 1: Self-adaptive scaling:

- Find the smallest nonnegative integer l^n such that $\alpha^{n+1} = u^{l^n}\gamma^n$ and update the non-shortest path flows using gradient projection $f_{\tilde{p}}^{n+1}(\alpha^{n+1}) = \max\left[0, f_{\tilde{p}}^n - \alpha^{n+1} \cdot g_{\tilde{p}}^n\right]$ satisfying

$$(2-\delta)\alpha^{n+1}(\mathbf{f}_{\tilde{p}}^n - \mathbf{f}_{\tilde{p}}^{n+1})^T(\mathbf{g}_{\tilde{p}}^n - \mathbf{g}_{\tilde{p}}^{n+1}) - (\alpha^{n+1})^2\left\|\mathbf{g}_{\tilde{p}}^n - \mathbf{g}_{\tilde{p}}^{n+1}\right\|^2$$
$$\geq \max\left\{0, \frac{(\alpha^{n+1})^2 - (\alpha^n)^2}{(\alpha^n)^2}\left\|e(\mathbf{f}_{\tilde{p}}^n, \alpha^n)\right\|^2\right\} \quad \text{(Condition 1)}$$

  Where, $\mathbf{f}_{\tilde{p}}^n$ and $\mathbf{g}_{\tilde{p}}^n$ represent the flow and gradient vectors of the non-shortest paths for O-D pair *rs* at iteration *n*; $g_{\tilde{p}}^n = c_{\tilde{p}}^n - c_{\bar{p}}^n$; $e(\mathbf{f}_{\tilde{p}}^n, \alpha^n) = \mathbf{f}_{\tilde{p}}^n(\alpha^n) - \mathbf{f}_{\tilde{p}}^{n+1}(\alpha^n)$.

- Update the shortest path flows $f_{\bar{p}}^{n+1} = D^{rs} - \sum_{\tilde{p} \neq \bar{p}} f_{\tilde{p}}^{n+1} \quad \forall rs \in RS$

Step 2: Selection of parameter γ^n

- If $\frac{1}{2}\alpha^{n+1}(\mathbf{f}_{\tilde{p}}^n - \mathbf{f}_{\tilde{p}}^{n+1})^T(\mathbf{g}_{\tilde{p}}^n - \mathbf{g}_{\tilde{p}}^{n+1}) - (\alpha^{n+1})^2\left\|\mathbf{g}_{\tilde{p}}^n - \mathbf{g}_{\tilde{p}}^{n+1}\right\|^2$ (Condition 2)
  $\geq \max\left\{0, \frac{(\alpha^{n+1})^2 - (\alpha^n)^2}{(\alpha^n)^2}\left\|e(\mathbf{f}_{\tilde{p}}^n, \alpha^n)\right\|^2\right\}$, then $\gamma^{n+1} = \min\left\{\frac{\alpha^{n+1}}{u}, \alpha^{\max}\right\}$.

  Otherwise, $\gamma^{n+1} = \alpha^{n+1}$.

Step 3: Convergence test

- If converged, STOP; Otherwise, $n = n+1$ and go to Step 1.

Figure 8.1 Self-Adaptive Gradient Projection (SAGP) Algorithm (Chen et al. 2012)

8.2 Disaggregate Self-Adaptive Gradient Projection (DSAGP)

In terms of exploring another assignment model, we consider the SAGP model by Chen et al. (2012) and fundamentally follow the SAGP model shown in Figure 8.1 with two modifications. First, instead of

applying an aggregate level step-size α^n at iteration *n* for all O-D pairs, we apply the SAGP at a disaggregate level for each O-D pair, α_{rs}^n for each O-D pair *rs*. When searching for an appropriate step-size, we are expecting a compact set of path alternatives limited by each O-D pair and a short search for the smallest integer *l* in Step 1, which may also depend on the O-D pair. Second, we apply a type of diagonalization technique in the SAGP model by maintaining the residual capacities for the estimated flows $\mathbf{f}^{n+1}$. Initially He et al. (2002) demonstrated the modified GLP model for asymmetric monotone cost functions, but Chen et al. (2012) introduced the modified GLP model for the non-additive traffic equilibrium problem. The proposed algorithm of disaggregate self-adjusting gradient projection (DSAGP) algorithm is as shown in Figure 8.2.

Step 0: Initialization

Enumerate paths using the hyperpath model

Set $n = 0$; $\alpha_{rs}^n > 0$; $\gamma_{rs}^n = \alpha_{rs}^n$; $u \in [0.5,1]$; $l_{rs}^n = 0$;

Step 1: Self-adaptive Scaling Procedure for each O-D

Step 1.1 Update the step-size parameter $\alpha_{rs}^{n+1} = u^{l_{rs}^n} \gamma_{rs}^n$

Step1.2 Find the shortest path and shift flow using the path-based gradient projection with α_{rs}^{n+1}

Step1.3 If Condition 1 in Figure 8.1 is satisfied, go to Step 2;

Otherwise, $l_{rs}^n := l_{rs}^n + 1$ then go to Step 1.

Step 2: Update γ_{rs}^{n+1}

According to Condition 2 in Figure 8.1, update γ_{rs}^{n+1} then go to Step 3.

Step 3: Convergence test

If convergence is satisfied, STOP and go to Step 1 until there are no more O-D pairs *rs*; Otherwise, $n = n + 1$ then go to Step 1.

Figure 8.2 DSAGP Algorithm

To measure convergence, we applied a variational inequality introduced by Smith (1983).

$$\min\ Z(\mathbf{f}) = \sum_{i=0}^{k} \max\left(0, (\mathbf{f} - \mathbf{v}^i)^T \cdot \mathbf{c}(\mathbf{f})\right)^{\beta} \tag{8.6}$$

Where, **f** is the feasible path flow vector; $\mathbf{v}^i$ stands for the extreme point vector of the feasible flow region; and **c**(**f**) represents the cost vector of feasible flow vector **f**.

For the stochastic UE model, the same features in Chapter 7.3 are applied in the DSAGP method. The cost vector **c**(**f**) carries the entropy term, and if no flows are assigned (the "sink-hole" effect), it will treat the "sink-hole" temporarily as applying one artificial unit of flow in the entropy term. In addition, the PSL is applied in the path enumeration stage before running the DSAGP algorithm.

8.3 Application

8.3.1 Example

For the simple model test, we applied the DSAGP in a path-based assignment on the same transit network as shown in Figure 6.2. A different objective function is used, following the variational inequality of Equation (8.6). This is necessary because the Lipschitz continuity condition is applied to asymmetric cost functions, although we assume a symmetric cost function utilizing diagonalization.

According to the VI objective function, Figure 8.3 shows the logarithmic objective value for the deterministic and stochastic DSAGP methods. Both models converge by iteration 12, although the convergence is not smooth in either the stochastic or deterministic models..

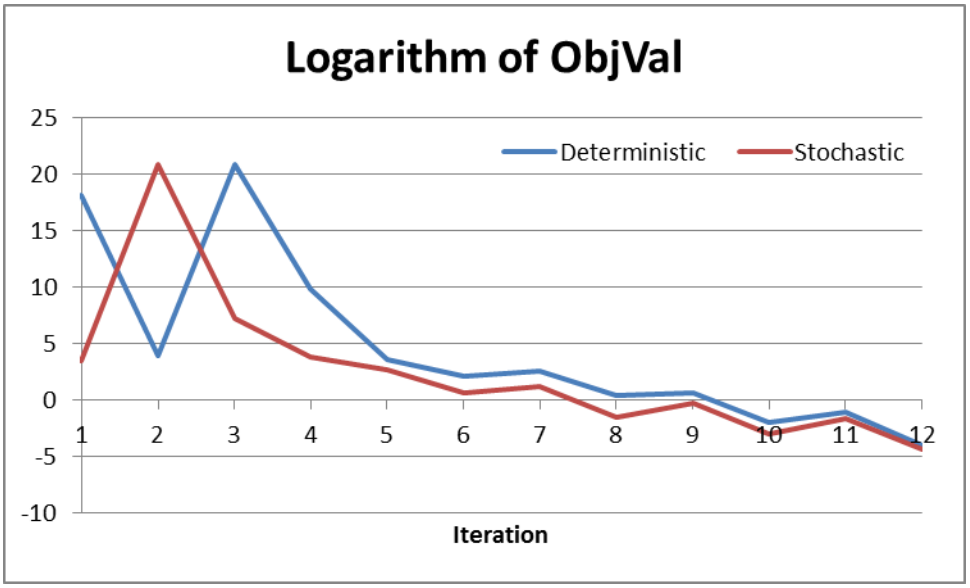


Figure 8.3 Logarithmic Objective Cost of DSAGP Model

More comparison results and analyses for this example network will be continued in Chapter 10.2.

9 COMPUTATIONAL MODEL STRUCTURE OF THE PROPOSED MODELS

9.1 Overall Structure of the Proposed Model

With four main theoretical transit assignment models, hyperpath-based and path-based assignments using the diagonalization technique and the full Hessian, and disaggregate self-adaptive gradient projection (DSAGP), the overall model structure with key functionalities is shown in Figure 9.1.

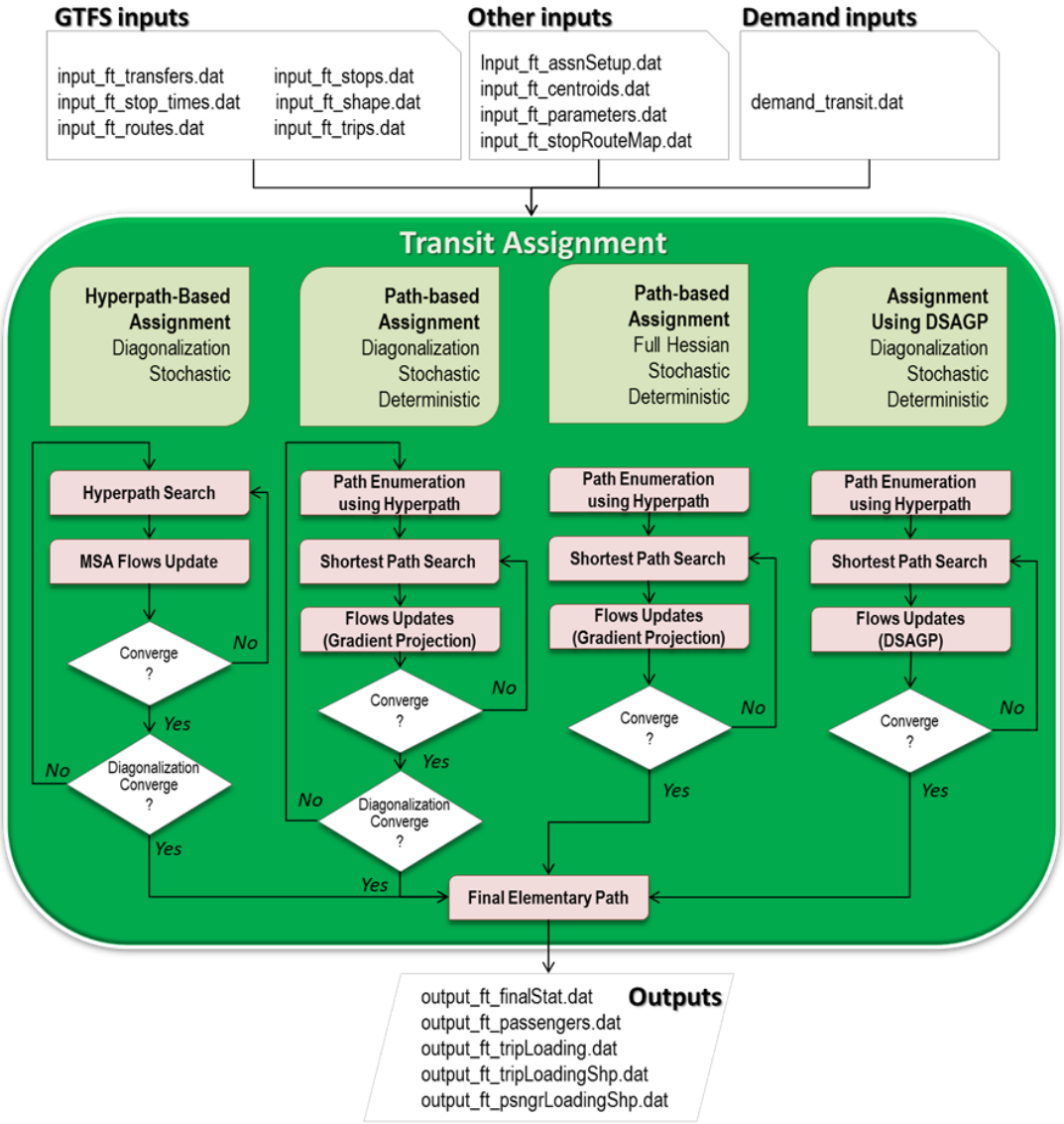


Figure 9.1 Transit Assignment Models and Flows of Input and Output Files

The input files are configured under Google's General Transit Feed Specification (GTFS), while the demand inputs and other miscellaneous inputs follow other specifications. GTFS inputs generally follow the same form of the original GTFS files, although subtle changes are applied (described in the following chapter). Other inputs are mainly prepared for maintaining the parameters of the proposed models and building access and egress links between stops and traffic analysis zone (TAZ) centroids. For demand inputs, we utilize trip files, giving origins, destinations, and an associated PAT or PDT for each trip.

Four main models – the hyperpath-assignment, two path-based models using diagonalization and full Hessian, and DASGP assignment – have their own functions as shown in Figure 9.1. First, for hyperpath-based assignment, the "Hyperpath Search" and "MSA Flows Update" are maintained in an inner iteration until the algorithm reaches the SUE or a predefined number of inner iterations. Second, the path-based and DSAGP assignment models have three main processes: "Path Enumeration", "Shortest Path Search", and "Flows Update". To consider an effective and efficient assignment, the path-based and DSAGP transit assignment models enumerate each elementary path, utilizing the existing hyperpath model and the path-size logit (PSL) for the SUE. After this initial path enumeration, the path-based and DSAGP methods iterate in the main loop (or inner loop only for the path-based model using diagonalization) between "Shortest Path Search" and "Flows Updates" until reaching a DUE or SUE.

For hyperpath- and path-based assignment using diagonalization, after the convergence of the inner loop, the model will check out the outer convergence, so called "diagonalization convergence". If the outer loop has not converged, the methods return to the inner loop. On the other hand, the path-based assignment using a full Hessian and DSAGP assignment models will directly check the convergence without measuring any inner iteration convergence. If a model converges, the assigned paths will be finalized through the "Final Elementary Path" component, which converts fractional passenger flows to integer flows on each elementary path, to be compatible with individual passengers in the demand input.

To create outputs after completing each transit assignment model, one more additional step in the box "Final Elementary Path" is required to create individual paths that are compatible with each individual traveler's tour. Above all, for the hyperpath-based assignment, a set of paths including their probabilities for the O-D-T tuple (or simply O-D pair, omitting the time interval T) are enumerated by the depth-first search using the property of Equations (6.4) ~ (6.7). The flows are estimated by multiplying the assigned demand with the probability of each path for each O-D-T. After rounding the fractional flows up or down to integer-numbered flows, integer flows are matched to the set of demands by O-D-T tuple. On the other hand, the path-based or DSAGP assignment models can directly utilize the enumerated path set to assign flows on each path. Similar to to the hyperpath-based assignment, floating-numbered flows on each path are rounded up or down to integer-numbered flows for each O-D-T tuple.

After achieving convergence or reaching the maximum number of iterations, the model creates several outputs: (1) overall passenger statistics such as travel time and distance, (2) passenger loading for each transit trip, (3) passenger travel outputs, (4) visualized outputs for each individual passenger showing the trajectory of passenger's trip including access, egress, and transfers, (5) visualized trip loading which depicts a different thickness according to the number of boardings at each stop along each transit vehicle trip, and (6) convergence and computational performance outputs.

9.2 Transit Network Structure

9.2.1 Nodes and Links

For building a transit schedule network, we categorize a transit network to links and nodes (See Figure 9.2). Nodes are separated into "OD node" and "Stop", and a unique ID for each stop and OD node is assigned independently, called "Local ID". For the overall assignment procedure one unique ID, "Global

ID", is assigned to each "OD Node" and "Stop". To match "Global ID" to "OD Node" or "Stop", a Node Map is structured, containing "Node Type" and "Local ID". The basic information for "OD Node" includes latitude and longitude, node type, and its local and global ID. "Stop" is made of GTFS stop information including latitude and longitude, stop type, stop name, and other column information in the GTFS stop file, *stops.txt*.

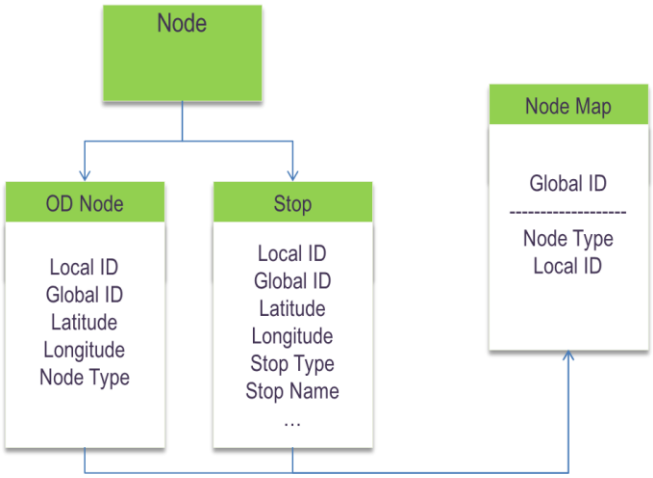


Figure 9.2 Node Map Structure

Input files required for configuring "OD Node" and "Stop" are *input_ft_centroids.dat*, *input_ft_nodes.dat*, and *input_ft_stops.dat*. The samples of these input files are shown in Figure 9.3. *input_ft_centroids.dat* has the information of TAZ geometry and centroids. *input_ft_nodes.dat* is generated automatically after reading *input_ft_centroids.dat* , keeping the origin and destination information according to the origin and destinations from the demand input files (*demand_transit.dat*) which includes trip-segment information, as shown in Figure 9.3(d). And *input_ft_stops.dat* has the same format of the GTFS file, *stops.txt*.

```
TAZ,RAD,LAT,LON,TYPE
466,East Sacramento,38.584892,-121.449277,od
467,East Sacramento,38.578626,-121.438435,od
468,East Sacramento,38.578497,-121.454678,od
```

(a) input_ft_centroids.dat

```
1,593,38.553878,-121.361869,od,East Sacramento
2,595,38.5513525,-121.3422251,od,Rancho Cordova
3,579,38.57139372,-121.313042,od,Rancho Cordova
4,570,38.58107195,-121.3264475,od,Rancho Cordova
```

(b) input_ft_nodes.dat

```
38.5782,344,-121.467,344,Buses head EB,F ST & 28TH ST (EB),0,2
38.5778,345,-121.466,345,Buses head EB,F ST & 29TH ST (EB),0,2
```

(c) input_ft_stops.dat

```
1000,4, 2, 1, 1, 593, 507057, 595, 506055, 3, 8, 4, 805, 832, 27.13, 0.35, 1, 0, 1
1000,4, 2, 2, 1, 595, 506055, 593, 507057, 3, 4, 8, 842, 909, 27.21, 0.35, 1, 0, 1
```

(d) demand_transit.dat

Figure 9.3 Input files for Stops and OD Nodes

In Figure 9.3, *input_ft_centroids.dat* has the columns of TAZ ID, Regional Area Description (RAD), Latitude(LAT), Longitude(LON), and node type, and *input_ft_nodes.dat* has the columns of internal TAZ, Latitude, Longitude, node type, and regional area description, *input_ft_stops.dat* consists of the columns of Latitude, Longitude, Direction of stop, Stop Name, Location Type, and Zone ID. For the passenger demand, *demand_transit.dat* has a sample number, person ID, tour number, tour half, trip number, origin

TAZ, origin parcel, destination TAZ, destination parcel, mode, origin purpose, destination purpose, departure time, arrival time, travel time, travel distance, expansion factor.

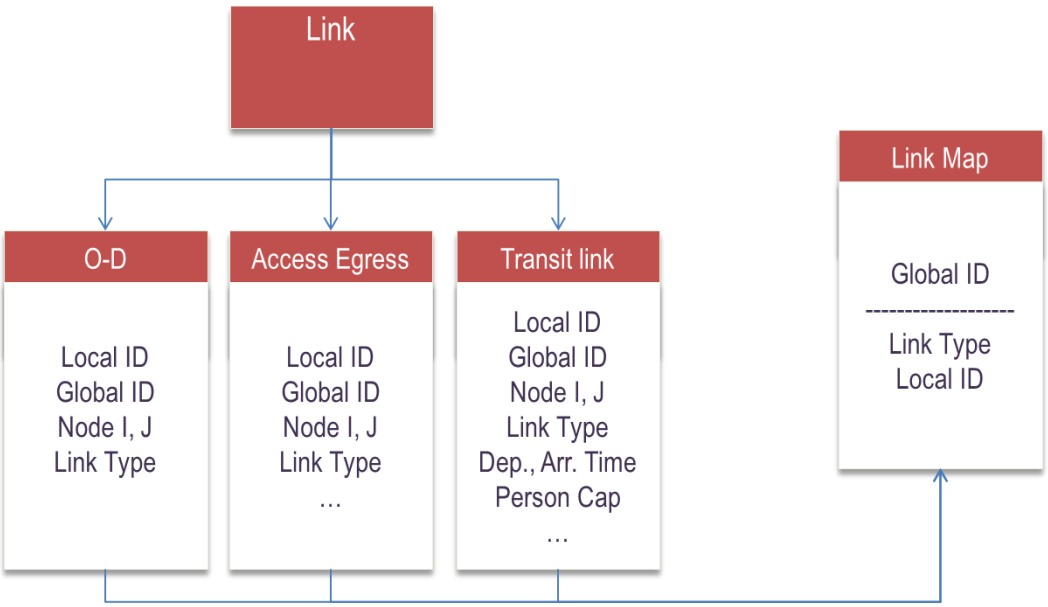


Figure 9.4 Link Map Structure

For configuring transit links, as shown in Figure 9.4, we also utilize the same mapping technique for the sub-links of “O-D”, “Access Egress” and “Transit link”. In the model run, we also use “Global ID” in “Link Map” which includes each link type, “Link Type” and “local ID” similar to Node Map. Three link types, “O-D”, “Access Egress”, and “Transit Link”, have their own local ID, “Node I, J” of each link, “Link Type” and “Global ID” mapped from “Link Map”. “Transit Link” is fundamentally structured by GTFS files which include *stop_time.txt*, *trips.txt*, *routes.txt*, and *transfers.txt*.

```
343868,13:54:00,13:54:00,3131,1,0,,,
343868,13:55:00,13:55:00,1860,2,0.198159,,,
343868,13:55:00,13:55:00,1861,3,0.328668,,,
343868,13:56:00,13:56:00,2020,4,0.464066,,,
343868,13:57:00,13:57:00,2021,5,0.732279,,,
```

(a) input_ft_stop_times.dat

```
a_1501,15,0,Watt I-80 Light Rail - Downtown,0,13234,1_merged_358352,341945,50
a_1501,15,0,Watt I-80 Light Rail - Downtown,0,13234,1_merged_358352,341946,50
a_1501,15,0,Watt I-80 Light Rail - Downtown,0,13234,1_merged_358352,341947,50
a_1502,15,0,Watt I-80 Light Rail - Downtown,0,13234,1_merged_358352,341948,50
```

(b) input_ft_trips.dat

```
INTERNATIONAL,3,000000,00FF00,SRTD,074,,74
MATHER FIELD,3,000000,FFFF00,SRTD,075,,75
ROSEMONT - LINCOLN VILLAGE,3,FFFFFF,800080,SRTD,072,,72
SUNRISE - CITRUS HEIGHTS,3,FFFFFF,FF0000,SRTD,021,,21
```

(c) input_ft_routes.dat

```
114,221,0,0.250764
114,222,0,0.084638
114,223,0,0.0612497
114,329,0,0.184326
```

(d) input_ft_transfers.dat

Figure 9.5 Input Files for Transit Schedule Links

For creating transit schedule links, *input_ft_stop_times.dat* (*stop_times.txt* in GTFS) consists of the foundation of the transit schedule links. In Figure 9.5, *input_ft_stop_times.dat* has the columns of trip ID, arrival time, departure time, stop ID, sequence ID, and mile post of each trip ID (or shape distance

traveled in GTFS). Consecutive stop-to-stop information forms the individual transit schedule links, and attributes of each link include trip ID, departure and arrival time of transit schedule link, stop I and J, and link distance along the mile post. *input_ft_trips.dat* simply follows the GTFS format (block ID, route ID, direction ID, trip head sign, trip type, shape ID, service ID, trip ID) except for capacity of each transit trip added in the last column, i.e., 50 persons. *input_ft_routes.dat* follows the GTFS format (route long name, route type, route text color, route color, agency ID, route ID, route URL, route short name) and *input_ft_transfers.dat* also follows the existing GTFS format (from stop ID, to stop ID, transfer type, min transfer time). However, we utilize the transfer distance (mile) instead of min transfer time in the last column and transfer time is estimated by using walking speed (3.0 mile/hour) in the assignment models.

9.2.2 Other Inputs

Running the model requires several additional input files for building a transit schedule network, generating outputs, and controlling the model as shown in Figure 9.6. Above all, *input_ft_stopRouteMap.dat* is critical for building better access and egress links considering the route number and head sign of each stop within a specific boundary, i.e., 1 mile radius distance. Figure 9.6(a) shows that each stop has a list of available routes and their directions. In the process of building access and egress links, overlapped stops including the same (route, head sign) pair are not considered in a set of access and egress links for each origin and destination. The file *input_ft_shapes.dat* is employed mainly for creating visualized outputs on Google Maps, such as each assigned passenger path and passenger loading on each trip. For controlling the overall model run, including the assignment configuration, the transit model requires *input_ft_assnSetup.dat*. The assignment model consists of the deterministic and stochastic model for the first hierarchy and, according to the first hierarchy, there are different assignment models: GP-Diag (gradient projection using diagonalization), GP-Hessian (gradient projection using full Hessian), DSAGP (disaggregate self-adaptive gradient projection) for both stochastic and deterministic models and HP (hyperpath-based MSA) for an additional stochastic model. In addition, the availability of

PSL for stochastic models and BIS (better initial solution) for the GP-Diag model are also categorized. For the detailed control of each model, such as numerous parameters for path generation, the flow shift in gradient projection, etc., *input_ft_parameters.dat* is needed. More detailed description of parameters will be discussed in the following chapter.

```
114|15,Buses head WB|
221|34,Buses head SB|
222|34,Buses head SB|
223|29,Buses head SB|34,Buses head SB|
```

(a) input_ft_stopRouteMap.dat

```
13290,38.495769,-121.480465,1,
13290,38.495769,-121.48035,2,
13290,38.494647,-121.480349,3,
13290,38.492562,-121.480335,4,
```

(b) input_ft_shape.dat

```
1. Assignment Model Setup
Deterministic (1) or stochastic (2): 1
Deterministic - GP-Diag (1) or GP-Hessian (2) or DSAGP (3): 3
Stochastic - GP-Diag (1) or GP-Hessian (2) or DSAGP (3) or Hyperpath(4): 2
Stochastic - Multinomial logit (1) or Path-size logit (2): 2
Better-feasible Solution (0 or 1): 0
```

(c) input_ft_assnSetup.dat

```
4. Hyperpath Search Parameters
Number of transfers: 15.0
Travel time: 1.0
Transfer time: 3.0
Waiting time: 3.0
Access time: 1.0
```

(d) input_ft_parameters.dat

Figure 9.6 Other Input Files

9.3 Parameters, Control, and Configuration Variables

To run the proposed models, appropriate parameters and control and configuration variables are required in the input file, *input_ft_parameters.dat*. These parameters and control and configuration variables are categorized in eight sub-sections: transit assignment configuration, time-expanded network configuration, hyperpath search configuration, hyperpath search parameters, path-overlapping control parameters, gradient projection parameters, capacity cost parameters, and DSAGP parameters.

9.3.1 Transit Assignment Configuration

The variables for the transit assignment configuration are shown in Table 9.1. First, to achieve the solution we separate two convergence types: iteration and relative gap. If iteration is chosen (1), "Outer iteration number" and "Inner iteration number" should be defined with appropriate values. Considering diagonalization iteration, "Outer iteration number" should be at least 2. On the other hand, if relative gap is chosen (2), "Relative gap for inner loop" and "Relative gap for outer loop" should be specified with appropriate user-defined values. Relative gap for inner and outer loop, defined *innerGap* and *outerGap* are estimated as follows.

$$innerGap = \frac{|Obj^{prev} - Obj^{current}|}{Obj^{prev}} \tag{9.1}$$

$$outerGap = \frac{\sum_{a \in A} \left| f_a^{prev} - f_a^{curr} \right|}{|A|} \tag{9.2}$$

Where Obj^{prev} and $Obj^{current}$ are the objective values of the previous and current iteration; f_a^{prev} and f_a^{curr} stand for flows on link *a* in the previous and current iteration, and |*A*| is the cardinality of links. To obtain a finer solution as shown in diagonalization step, we can also set a different *outerGap* as $\max \left| f_a^{prev} - f_a^{curr} \right| / f_a^{prev}$. If the *innerGap* and *outerGap* satisfy the predefined criteria such as 0.0001

and 0.001 shown in Table 9.1, the proposed criteria will finalize the assignment process. The maximum outer and inner iterations are set to the predefined bounds by “Max outer iteration for relGap” and “Max inner iteration for relGap”, and those are active if the relative gap, *innerGap* or *outerGap* is not satisfied. We note that the path-based model using a full Hessian only utilizes “Inner Iteration Number”, *innerGap* and “Max inner iteration for relGap” and the DSAGP model uses the “Outer iteration number” and an individual convergence criterion, “epsilon” in Table 9.8 for assignment configurations.

Table 9.1 Transit Assignment Configuration

Transit Assignment Control Variables	Description	Probable Values
Iteration (1) or relative gap (2)	To decide iteration or relative gap for running a model	1 or 2
Outer iteration number	Number of outer iterations (or diagonalization iteration)	4
Inner iteration number	Number of inner iterations	10
Relative gap for inner loop	Relative gap for inner loop	0.0001
Relative gap for outer loop	Relative gap for outer loop	0.001
Max outer iteration for relGap	Maximum outer iteration for relative gap (2)	3
Max inner iteration for relGap	Maximum inner iteration for relative gap (2)	20

9.3.2 Time-expanded Network Configuration

To keep a manageable network size including reasonable passenger path generation, three configuration variables are considered; “Max time prism from departure time”, “Access and egress distance to stop,” and “Number of max access nodes”. “Max time prism from departure time” controls the temporal path search area, to exclude abnormal paths beyond the Max time prism, or simply the maximum total travel time boundary from origin to destination. For example, in Table 9.2, 200 minutes is set as an acceptable time prism from PDT at origin or to PAT at destination. “Access and egress distance to stop” is for collecting possible stops within the defined Euclidean distance or radius, and the total number of stops is

no more than "Number of max access nodes". For instance in Table 9.2, if the number of stops for access and egress within the defined Euclidean distance (1.2 miles) are more than 8, only the first 8 stops (ordered by increasing distance) are included in the set of stops to build access and egress links. To generate access and egress links, *input_ft_stopRouteMap.dat* is utilized to consider stops with unique route and direction by head sign for access and egress stop alternatives.

Table 9.2 Time-expanded Network Configuration

Time-expanded Network Configuration	Description	Probable Values
Max time prism from departure time (min)	Maximum buffer time from origin to destination	200
Access and egress distance to stop (mile)	Euclidean radius for access and egress distance	1.2
Number of max access nodes	Maximum number of nodes considered for access	8

9.3.3 Hyperpath Search Configuration

To generate a hyperpath, first we need to set the boundary of PAT or PDT including a set of transit trip alternatives arriving at the destination before the PAT or leaving from the origin after the PDT, given the time windows in Table 9.3. We assume that each passenger has his/her own buffer regarding PAT or PDT, such as 30 minutes earlier than the PAT or later than the PDT. If a small PAT boundary is set, such as 10 minutes, the possibility of having transit alternatives at the alighting stop is decreased, especially for the stops with less frequent transit service. In Table 9.3, "Time Interval Bin Resolution" represents the internal time interval for aggregating the demand. For example, if there is a passenger with PAT of 9:00 AM and another passenger with PAT 8:57 AM , both are loaded in the PAT 9:00AM bin within the predefined PAT boundary, i.e., 30 minutes (8:30AM ~ 9:00AM).

Table 9.3 Hyperpath Search Configuration

Hyperpath Search Configuration	Description	Probable Values
PAT time boundary (min)	PAT time buffer to the latest PAT time at Destination	30
PDT time boundary (min)	PDT time buffer from the earliest PDT time at Origin	30
Time Interval Bin Resolution (min)	Time interval for aggregating demand according to the PAT or PDT	5

9.3.4 Hyperpath Search Parameters

The hyperpath is basically structured on the logit model, so that calibrated parameters can be used. The parameters used for creating a weighted cost or (dis)utility are shown in Table 9.4: “Number of transfers”, “Travel, Transfer, Waiting, Access and Egress time”, “Early arrival for PAT”, and “Late departure for PDT”. “Logit model dispersion factor (Theta)” is also included in the set of parameters. In addition, to generate a hyperpath, two alternative acceptance boundaries are included in the set of parameters; “Alternative acceptance boundary” and “Probability accepted for alternative”. “Alternative acceptance boundary” manages a set of alternatives, effectively decreasing the log-sum weighted cost, which includes alternatives decreasing the weighted cost more than the specified values. “Probability accepted for alternative” has a similar role as managing an alternative set together with “Alternative acceptance boundary”. Each alternative with probability more than the predefined probability is included in the alternative set at each stop.

Table 9.4 Hyperpath Search Parameters

Hyperpath Search Parameters	Description	Probable Values
Number of transfers	Weighting parameter for each transfer	15.0
Travel time	Weighting parameter for transit travel time	1.0
Transfer time	Weighting parameter for transfer time	3.0
Waiting time	Weighting parameter for waiting time	3.0
Access time	Weighting parameter for access time	1.0
Egress time	Weighting parameter for egress time	1.0
Early arrival for PAT	Weighting parameter for early arrival to the latest PAT time	1.2
Late departure for PDT	Weighting parameter for late departure from the earliest PDT time	1.2
Logit sensitivity (Theta)	Dispersion parameter of logit model	- 0.8
Alternative acceptance boundary in hyperpath	Lower bound of contribution in Log-sum for adding an alternative	0.01
Probability accepted for alternative	Lower bound of probability for each alternative	0.001

9.3.5 Path-Overlapping Control Parameters

With the logit-based hyperpath, overlapping of alternative paths should be considered. As discussed in the model development in Chapter 0, the path-size logit (PSL) model is employed for overcoming the path overlapping problem. The typical parameters in PSL are shown in Table 9.5. Path-size logit (PSL) beta is a parameter in the logit function, Equation (9.3), and the path-size logit (PSL) gamma is the parameter in path-size (PS), shown in Equation (9.4).

Table 9.5 Path-overlapping Control Parameters

Path-Overlapping Control Parameters	Description	Probable Values
Path-size logit(PSL) beta in logit	Parameter beta in upper logit model	1.0
Path-size logit(PSL) gamma in PS	Parameter gamma in lower Path Size (PS) model	2.0

$$P_p = \frac{\exp(C_p + \beta \cdot \ln PS_p)}{\sum_{j \in Q} \exp(C_j + \beta \cdot \ln PS_j)} \tag{9.3}$$

$$PS_p = \sum_{a \in A_p} \frac{l_a}{l_p} \cdot \frac{1}{\sum_{j \in Q} \left(\frac{l_p}{l_j} \right)^{\gamma} \delta_{aj}} \tag{9.4}$$

9.3.6 Gradient Projection Parameters

When we consider the path-based assignment model, we have a flow adjusting parameter for controlling the shift of flows in the gradient projection, "Flow-shift alpha" in Table 9.6 and α in Equation (9.5).

Table 9.6 Gradient Projection Parameters

Gradient Projection Parameters	Description	Probable Values
Flow-shift alpha	Parameter alpha in gradient projection	1.0

$$f_{\tilde{p}}^{n+1} = \max\left[0,\ f_{\tilde{p}}^{n} - \alpha \left(\frac{\partial^2 \tilde{Z}}{\partial f_{\tilde{p}}^{n\,2}} \right)^{-1} \cdot (d_{\tilde{p}}^{n} - d_{\bar{p}}) \right] = \max\left[0,\ f_{\tilde{p}}^{n} - \alpha \cdot g^n / h^n \right] \tag{9.5}$$

9.3.7 Capacity Cost Parameter

Regarding the capacity cost functions introduced in Chapter 5.2.4, “Capacity Alpha” maintains the steepness of the monotone capacity cost and “Capacity Beta” decides the relative residual capacity of the power function in (9.7). “Capacity Cost Model(exponential: 1, power: 2)” specifies a capacity function in the model run between the exponential and the power functions. Exponential and power functions are shown in Equations (9.6) and (9.7).

Table 9.7 Capacity Cost Parameters

Capacity Cost Parameter	Description	Probable Values
Capacity Alpha	Parameter alpha in capacity cost	3.0
Capacity Beta	Parameter beta in capacity cost	0.7
Capacity Cost Model(exponential: 1, power: 2)	Capacity cost function type (1:exponential function ; 2: power function)	1

$$c_{ab}^{cap} = \frac{f_{ab}}{\max[\,0, r_b - \sum_{o_{kb}^{m_k m_b} < o_{ab}^{m_a m_b}} f_{kb}\,]} \exp\left(\alpha \cdot (f_{ab} - \max[\,0, r_b - \sum_{o_{kb}^{m_k m_b} < o_{ab}^{m_a m_b}} f_{kb}\,]) \right) \tag{9.6}$$

$$c_{ab}^{cap} = \frac{f_{ab}}{r_b} \max\left[0,\, f_{ab} - \beta \cdot r_b\right]^{\alpha} \tag{9.7}$$

9.3.8 DSAGP Parameters

In terms of applying the DSAGP, we need to consider parameters to control the DSAGP model in Table 9.8. “Initial alpha[>0]” represents the initial step-size of the gradient projection model used in DSAGP which allows any positive value, varying from 1.0 to 99999, and “Max alpha-parameter” maintains the maximum flow shift by this upper bound of the step-size. “Parameter *u* in alpha[0.5<=*u*<=1]” is a fixed

parameter and critical to determine step-size, and "Parameter delta in condition 1[0<delta<1]" is a scalar for the continuous condition 1 used in the DSAGP algorithm. "Max *l*-parameter" controls the number of iterations for searching for the appropriate step-size α. Finally, "Epsilon for convergence" stands for the convergence criterion of the DSAGP algorithm in which to estimate the convergence value, where Equation (9.8) is utilized.

Table 9.8 DSAGP Parameters

DSAGP Parameters	Description	Probable Values
Initial alpha[>0]	Initial step-size α in gradient projection	9999
Parameter *u* in alpha[0.5<=*u*<=1]	Given parameter *u* in $\alpha^{n+1} = u^{l^n} \gamma^n$	0.8
Parameter delta in condition 1[0<delta<1]	Scalar parameter for the continuous condition	0.5
Max *l*-parameter	Maximum iteration of searching alpha	1000
Max alpha-parameter	Maximum value of step-size	9999
Epsilon for convergence	Convergence criterion	0.001

$$\varepsilon = \sum_{i=0}^{k} \max\left(0, (\mathbf{f} - \mathbf{v}^i)^T \cdot \mathbf{c}(\mathbf{f})\right)^{\beta} \tag{9.8}$$

9.4 Outputs

The proposed hyperpath- and path-based model runs create several output files; overall travel time and distance, computation performance, assigned passengers and trip loadings. The file *output_ft_finalStat.dat* shows the overall statistics including total and average travel time and distance as shown in Table 9.9 and Table 9.10.

Table 9.9 Outputs of Travel Time

Travel Time Outputs	Description
o-d travel time (min)	Travel time origin to destination
travel time (min)	Travel time excluding access and egress time
in-veh travel time (min)	Transit-only in-vehicle travel time
transfer time (min)	Transfer time
waiting time (min)	Waiting time
access time (min)	Access time from origin to boarding stop
egress time (min)	Egress time from alighting stop to destination
number of transfer	Number of transfers

Table 9.10 Outputs of Travel Distance

Travel Distance Outputs	Description
o-d travel dist (mile)	Travel distance from origin to destination
travel dist (mile)	Travel distance excluding access and egress distance
in-veh travel dist (mile)	Transit-only in-vehicle travel distance
transfer dist (mile)	Transfer distance
access dist (mile)	Access distance from origin to boarding stop
egress dist (mile)	Egress distance from alighting stop to destination

For the computational performance, the proposed models give *output_ft_AssnConv.dat* (see Figure 9.7) including iterative objective values and the path, capacity, and entropy cost for the path-based model, PSL, and the relative gap which is defined by $innerGap = \left| \left(Obj^{prev} - Obj^{curr} \right) / Obj^{prev} \right|$. The file *output_ft_passengers.dat* shows the trajectory of each passenger which includes passenger ID, mode type, from time, origin, destination, a series of boarding stops, trips, and alighting stops, and pair of (access, transfer, and egress time). If there is no transfer, then the last column only includes access and egress

times. The file *output_ft_tripLoading.dat* represents the passenger loading on each trip by counting the number of boarding passengers at each stop along the trip.

OutIter	InIter	ObjValue	PathCost	CapCost	EntropyCost	PSL	innerGap
0	1	413698	413475	63.1582	160.439	0	1.0
0	2	413756	413472	124.343	159.609	0	0.000139333
0	3	413695	413472	63.1582	159.606	0	0.000147889
0	4	413695	413472	63.1582	159.603	0	1.24413e-008
0	5	413695	413472	63.1582	159.600	0	1.24408e-008

(a) output_ft_AssnConv.dat

```
2399.4211  3  596  595  406  2885  344016  2892     1.95548,4.43088
3740.4211  3  596  595  406  2885  344016  2892     1.95548,4.43088
1930.3211  3  565  595  397  5357,4210  344077,343992  3852,3861  15.2635,2.25165,4.44287
3271.3211  3  565  595  397  5357,4210  344077,343992  3852,3861  15.2635,2.25165,4.44287
```

(b) output_ft_passengers.dat

tripID	stopID	numBoardings
343868	3131	0
343868	1860	1
343868	1861	5
343868	2020	10

(c) output_ft_tripLoading.dat

Figure 9.7 Outputs of Convergence, Passengers, and Trip Loading

For more specific and descriptive outputs considering the geographical area, the assignment models produce two visualization outputs, especially using the Google Maps API as shown in Figure 9.8 and Figure 9.9. *output_ft_tripLoadingShp.dat* generates every trip html trajectory with a Java Script call of the

Google Maps API. According to the amount of passenger loading on each transit trip (or vehicle), the thickness of the trip is changed along the trajectory of trip, which is also shown in Figure 9.8.

Figure 9.8 output_ft_tripLoadingShp.dat

The file *output_ft_psngrTripLoading.dat* includes every passenger's html trajectory with Java Script assigned on the transit network. For this visualization, *input_ft_shapes.dat* is mainly used for producing those files. For each assigned passenger, Figure 9.9 shows the trajectory including access and egress in a blue color and transfer in a green color. Other transit trajectories are shown in a red color. These visualized outputs allow to checking if any passenger's trip is appropriate and will help to provide better transit service planning.

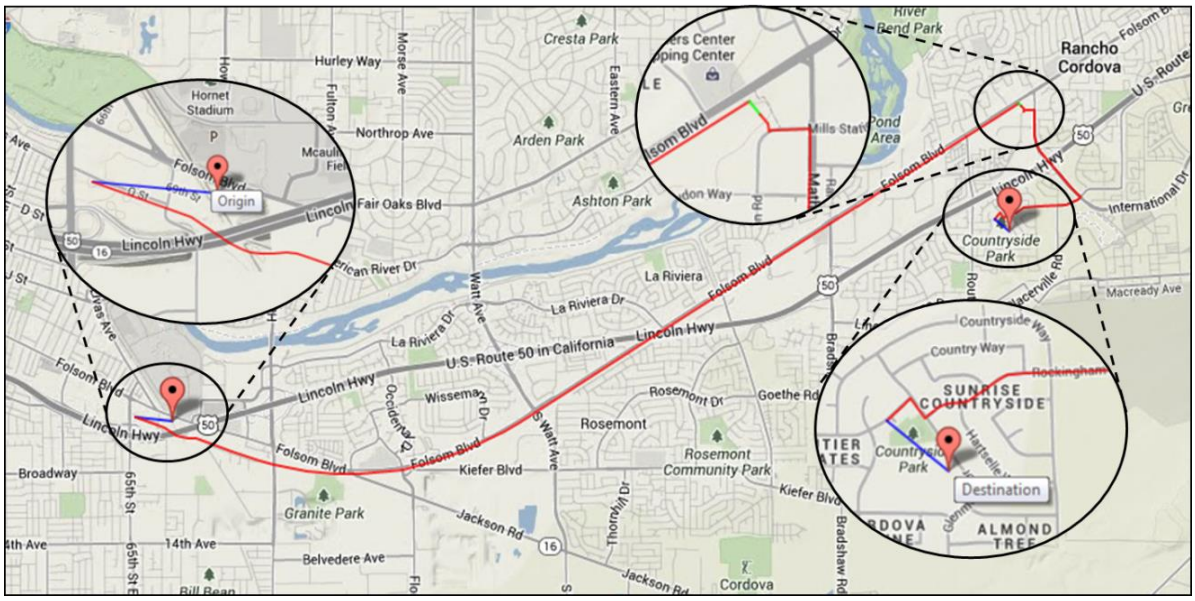


Figure 9.9 output_ft_psngrLoadingShp.dat

10 APPLICATIONS AND RESULTS

10.1 Application Environment

Through this study, we developed three main transit assignment models. One is a hyperpath-based model (HP) using a method of successive average (MSA); the second model is a path-based model using gradient projection (GP); and the third model utilizes the self-adaptive gradient projection technique. For the GP model, we also tested a better initial solution (BIS) model. The hyperpath model provides the main path search in HP and provides the path enumeration in the GP and DSAGP models. The proposed HP, GP, and DSAGP models are tested on an Intel Core i3 2.4GHz with 4GB RAM, and Visual Studio 2008 (Win32 and .Net 2.0) is used for compiling and computation with the developed C++ code. For conceptual verification of the proposed models, we used a simple network from previous chapters, and the proposed models are also tested on a partial Sacramento transit network including Downtown, East Sacramento, and Rancho Cordova.

In more detail, the proposed models and their solution methods are characterized in Table 10.1. For stochastic behavior, *first*, the proposed models are fundamentally categorized into deterministic and stochastic models, in which the stochastic models include an entropy-based dispersion and employ the path-size logit (PSL) model to correct for path overlaps. For the solution methods, *second*, we consider three basic models: a path-based assignment model using gradient projection (GP), a self-adaptive gradient projection using the pseudo Lipschitz continuity condition (DSAGP), and the method of successive averages (MSA) using a hyperpath construct (HP). *Third*, each basic model is broken to sub-models. GP has two sub-models, GP-Diag and GP-Hessian, according to whether it uses a diagonalization technique for the non-separable and asymmetric cost function or a full Hessian scaling matrix. Typically, GP-Diag uses diagonal cells of the first derivative of the Jacobian matrix. On the other hand, GP-Hessian does not use diagonalization but rather the full matrix of the first derivatives of the Jacobian matrix.

Another model using diagonalization is HP-Diag, which uses MSA for passenger assignment. The diagonalization model, GP-Diag, is further branched to GP-Diag-BIS as utilizing a better initial solution heuristic to find a minimum cost solution with priority of boarding on a capacitated transit schedule network. In the final case, the DSAGP model utilizes a pseudo Lipschitz continuity condition to find an optimized flow shift parameter value for each origin-destination pair and for every time interval, in which flows are shifted by gradient projection with the optimized step-size.

Table 10.1 Category of the Proposed Models

Category	Solving Models	Model Name	Notes
Deterministic	Path-based Model Using Gradient Projection (GP)	GP-Diag	Diagonalization
		GP-Diag-BIS	Diagonalization and Better Initial Solution (BIS) heuristic
		GP-Hessian	Hessian
	Self-Adaptive Gradient Projection (SAGP)	DSAGP	Disaggregate Self-Adaptive Gradient Projection
Stochastic using path-size logit (PSL)	Path-based Model using Gradient Projection (GP)	GP-Diag	Diagonalization
		GP-Diag-BIS	Diagonalization and Better Initial Solution (BIS) heuristic
		GP-Hessian	Hessian
	Self-Adaptive Gradient Projection (SAGP)	DSAGP	Disaggregate Self-Adaptive Gradient Projection
	Method of Successive Averages (MSA) using Hyperpath (HP)	HP-Diag	Diagonalization

10.2 Simple Network Test

For the overall model test, first we revisit the simple network as shown in Figure 10.1. We assume that there are four routes running over 11 transit stops (numbered from 1 to 11) between two origins (O_1 and O_2) and two destinations (D_1 and D_2). All links between two consecutive stops are transit links except for four transfer links (9,2), (5,10), (10,5), and (11,8). The other links connecting from/to origins or destinations are access and egress links. For the schedule of transit service, we simply assume one schedule per each route which is shown on each stop. These times are assumed to be either scheduled arrival or scheduled departure times). Numbers over links show the walking times of access, egress, and transfer. Demand is shown below the network. Total origin demands are 15 and 16 for origins O_1 and O_2, respectively. Total destination demands are 19 and 12 for destination D_1 and D_2, respectively. Capacity is assumed to be 10 units for Routes 1, 2, and 3, and Route 4 has a capacity of 20 units.

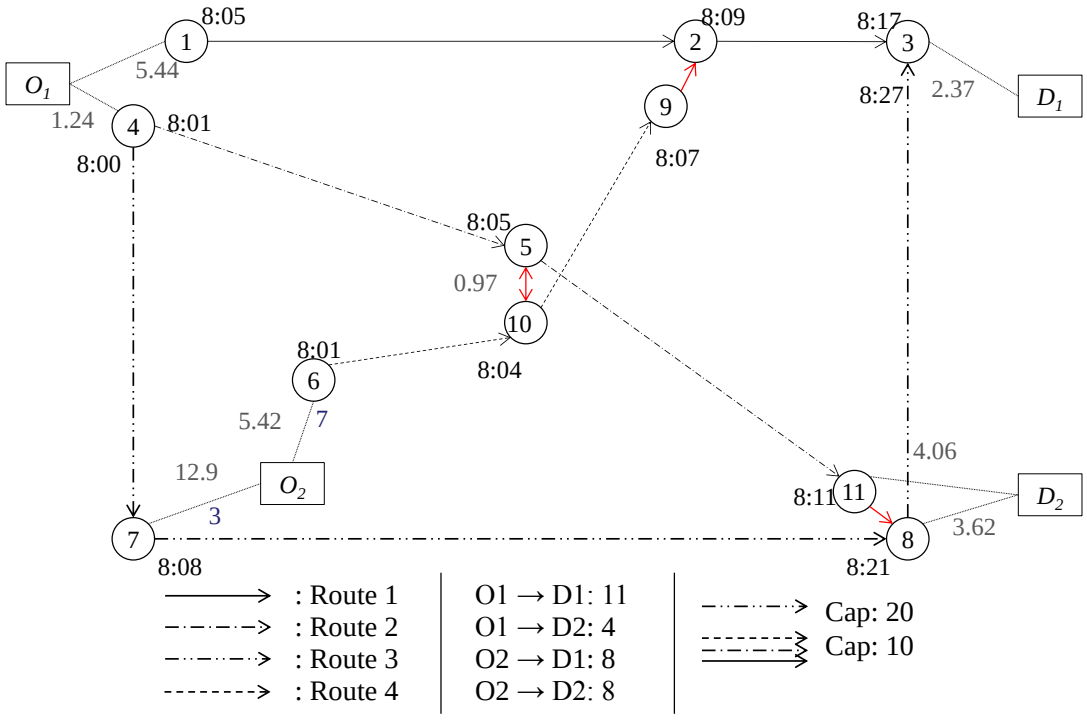


Figure 10.1 Simple Network with Demand and Capacity

The initial flows by all-or-nothing assignment are shown in Figure 10.2, where, the numbers over the bold arrows between origins and destinations represent loaded. This figure also shows a set of preferred paths by passengers between origins and destinations. Priority consists in the boarding competition between on-board passengers and boarding passengers. For example, passengers using link (1,2) and link (4,5) take priority of boarding on link (2,3) and (5,11) over the passengers from link (10,9) and link (6,10), because they are on-board passengers on Routes 1 and 2, respectively.

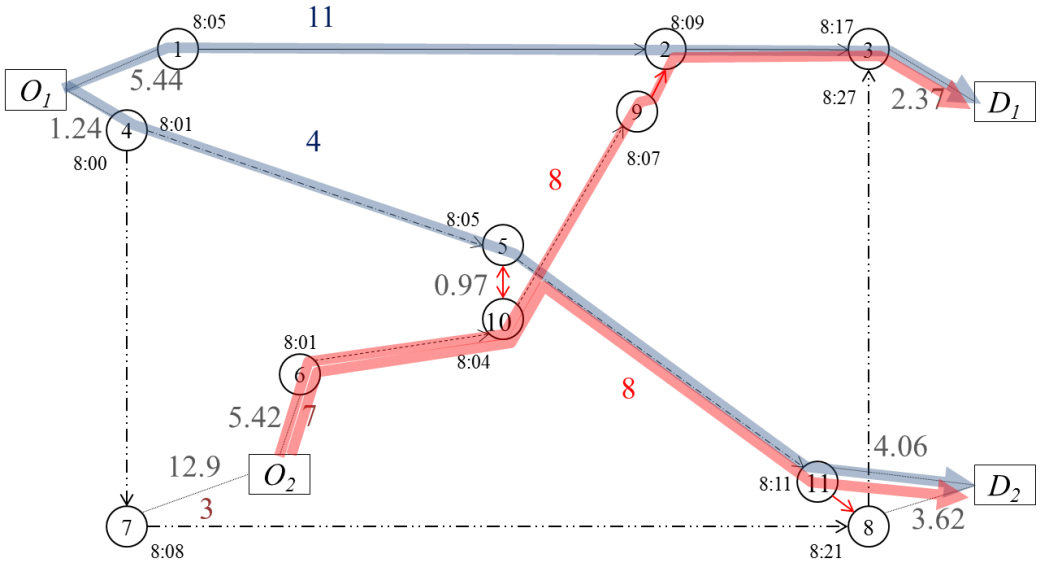


Figure 10.2 Initial Loading by All-or-Nothing

From the all-or-nothing loading, we detect the congested links as (1,2), (2,3), (6,10) and (5,11). Considering the capacity of each link and the passenger priority, we can estimate a feasible solution from the proposed deterministic better initial solution (D-BIS) in the previous chapter. When we apply D-BIS, the solution is given in Figure 10.3.

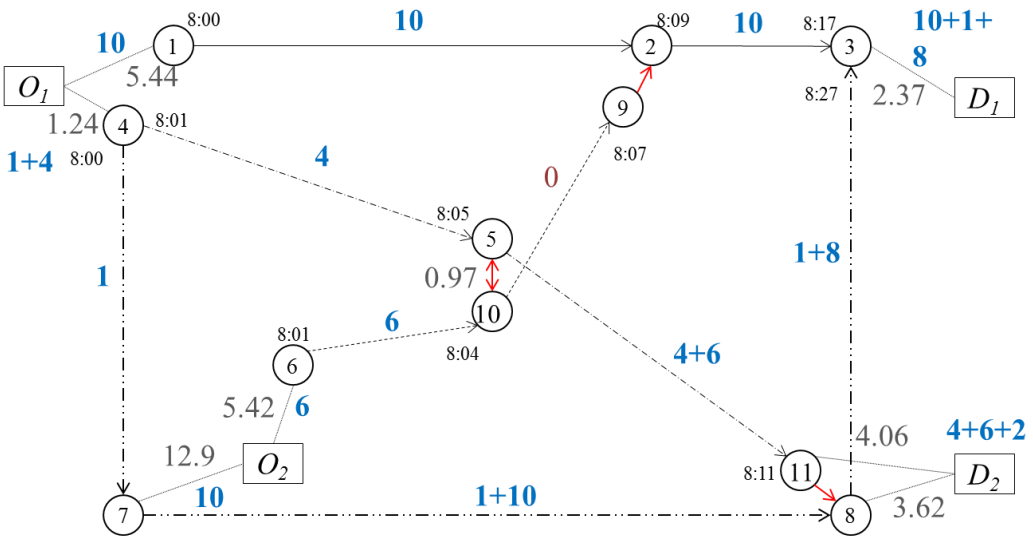
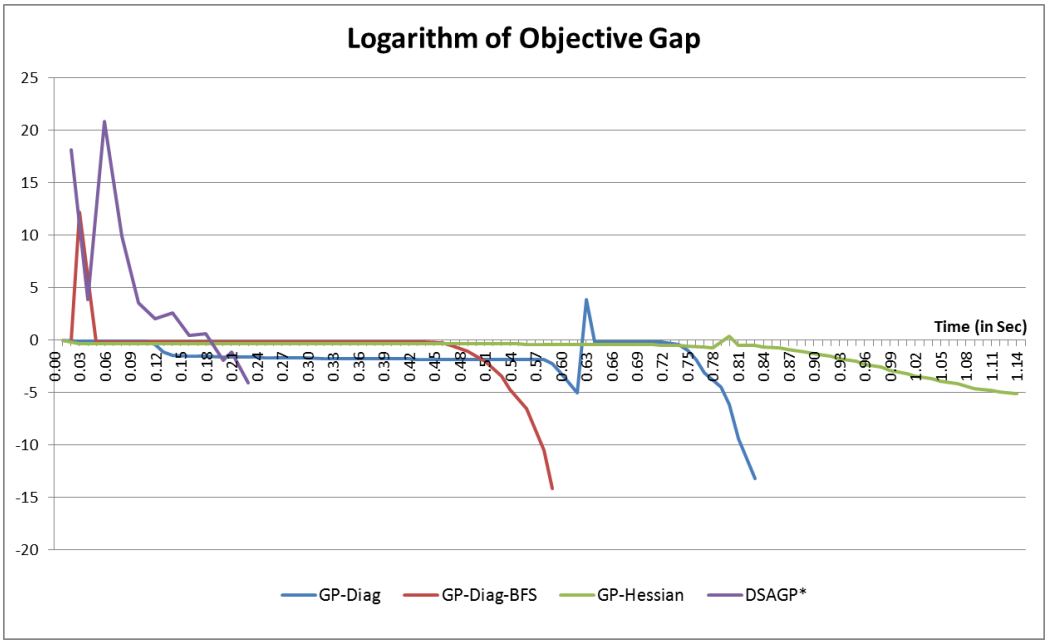


Figure 10.3 Loaded Flows by Deterministic Better Initial Solution (D-BIS)

With this initial feasible solution shown in Figure 10.3, convergence, performance and solutions can be achieved by applying the proposed deterministic models on the simple network as shown in Figure 10.4.



*: DSAGP applies VI objective function

Figure 10.4 Logarithmic Objective Gap Values of the Proposed Deterministic Models

Figure 10.4 represents the convergence results among the proposed models. Above all, we applied convergence criteria: 10^{-4} for DSAGP, 10^{-3} and 10^{-5} for the outer (diagonalization) and inner loops of GP-Diag, GP-Diag-BIS, and 10^{-5} for GP-Hessian model, in which GP-Diag and GP-Diag-BIS show the inner objective gap values. Also, the convergence rate of DSAGP does not include the internal iterations necessary to estimate the step-size α, so that the performance of DSAGP is underestimated and may not guarantee faster performance against other methods in a real application.In Figure 10.4, GP-Diag and GP-Diag-BIS show different convergence rates, with GP-Diag slower than GP-Diag-BIS, although both show peaks around 0.03 seconds and 0.63 seconds for GP-Diag-BIS and GP-Diag, respectively. These peaks are related to a sudden capacity cost increase, typically at a starting point of a new diagonalization iteration. On the other hand, GP-Hessian shows a gradual convergence rate requiring around 1.14 seconds (77 iterations) to converge.

Although it does not demonstrate similar performance on a large transit network, we investigated the performance of the proposed models in Figure 10.5. The performances show the compatible results to the convergence rate in Figure 10.4, in the order from quickest to slowest of DSAGP, GP-Diag-BIS, GP-Diag, and GP-Hessian. However, the performance of DSAGP may show the different performance over a real large transit network according to its internal iterations of estimating an optimal step-size parameter.

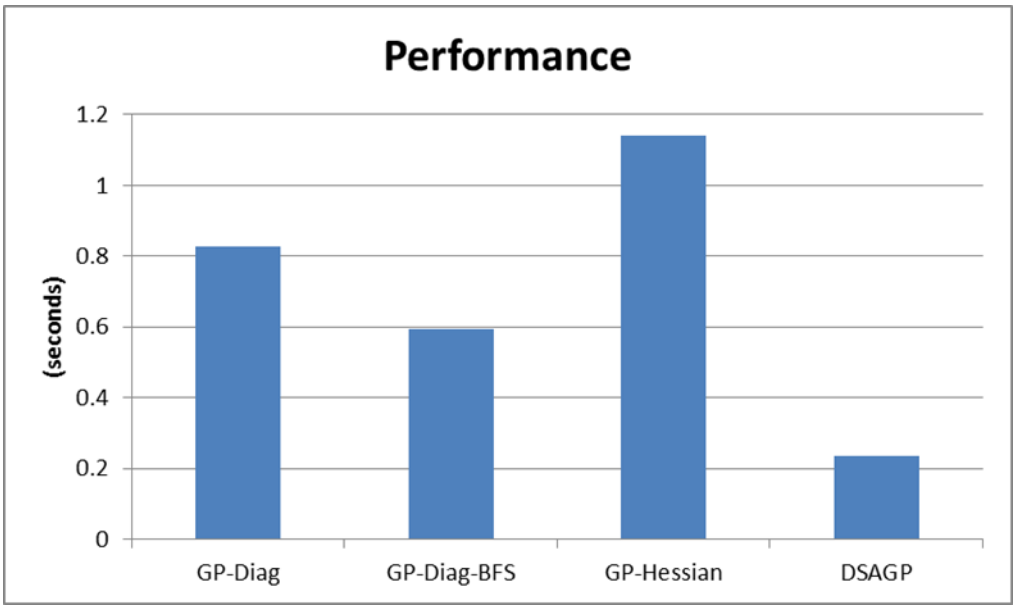


Figure 10.5 Performance of the Proposed Deterministic Models

As solutions, the passenger loadings of the proposed models are shown in Figure 10.6. The passenger loadings at each stop along each route (or trip) show exactly the same results, including the same violations at Stops 1 and 2 of Route 1 and Stop 5 of Route 2. These violations show the typical output of a "soft-capacity" model, which allows loading passengers over the defined capacity of a vehicle. Also, the passenger loadings are shown in integer-value results, although it is originally in a floating number. This is because the flows with floating numbers are rounded up/down to match the individual passenger trip in the tour-based demand input.

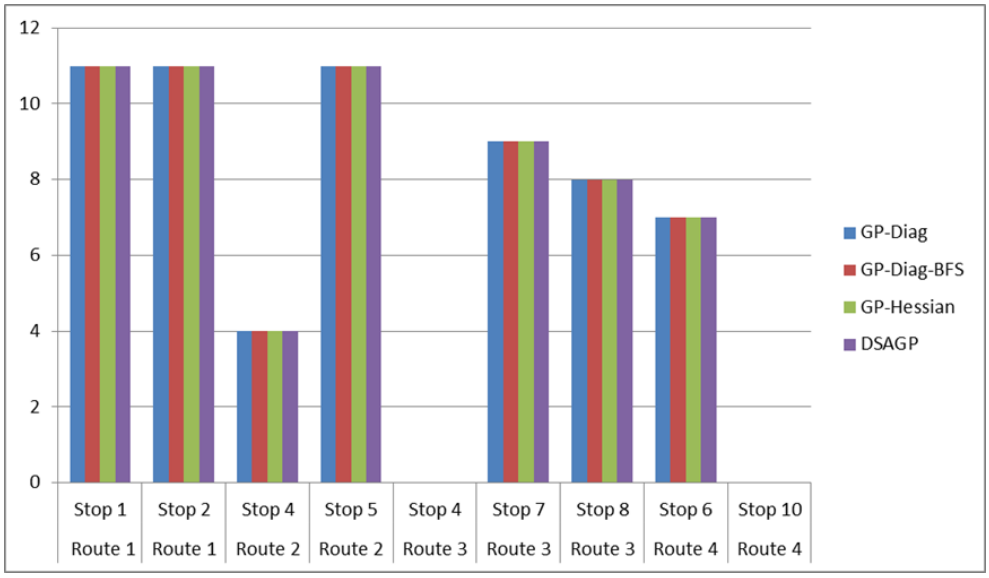
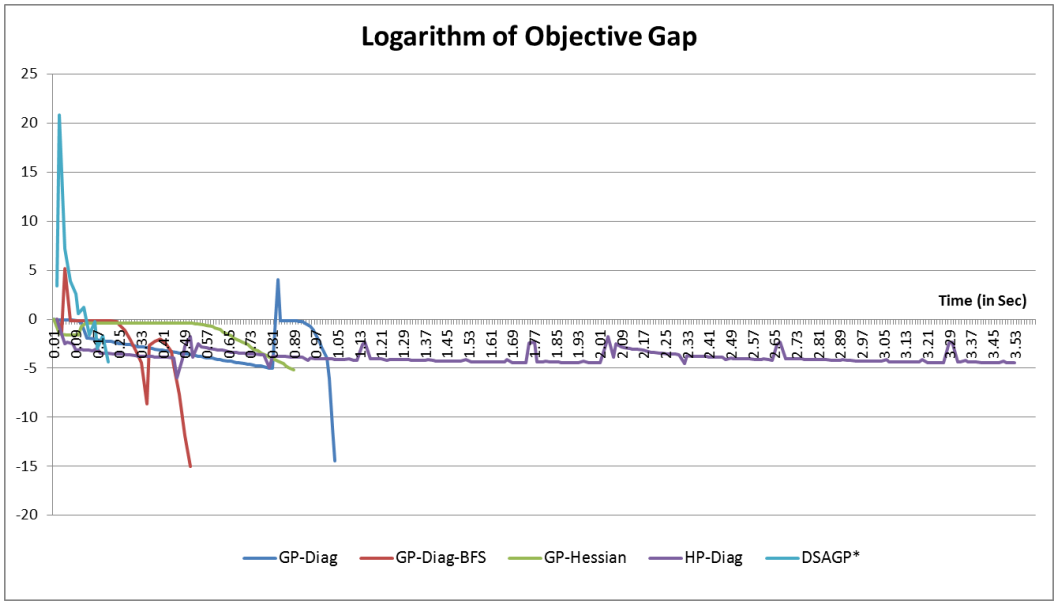


Figure 10.6 Passenger Loadings of the Proposed Deterministic Models

For the stochastic models, we have added HP-Diag with the other GP and DSAGP models for comparing the convergence rates, shown in Figure 10.7, and 10^{-5} and 10^{-3} convergence criteria are applied for inner and outer loops, respectively.

On the number of iterations, noticeably, HP-Diag has fewer but longer iterations, as it reaches its convergence earlier around at 0.45 seconds. The algorithm keeps moving on to the next diagonalization iteration, under the requirement of "at least" two iterations the same between the GP-Diag and GP-Diag-BIS models. HP-Diag shows gradual increases and decreases until the model converges. Different from Figure 10.4, GP-Hessian and GP-Diag-BIS show a little faster convergence, but GP-Diag is slower than the paired deterministic GP-Diag as shown in Figure 10.4. Also, ranking the performance on the convergence shows the order of DSAGP, GP-Diag-BIS, GP-Hessian, GP-Diag, and HP-Diag from the lowest to highest computation performance.



*: DSAGP applies VI objective function

Figure 10.7 Logarithmic Objective Gap Values of the Proposed Stochastic Models

In terms of performance (see Figure 10.8), these methods follow the same order as the convergence rates in Figure 10.7.

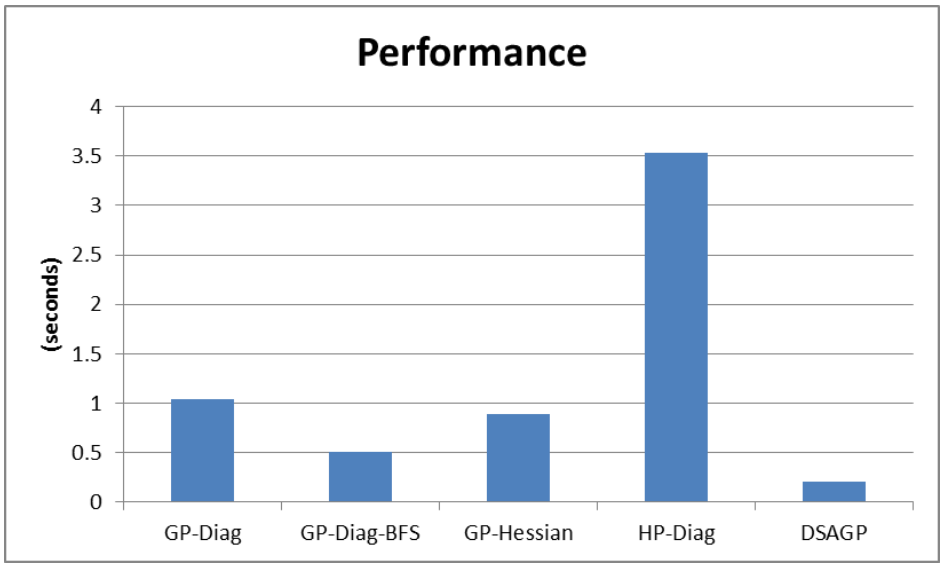


Figure 10.8 Performance of the Proposed Stochastic Models

Regarding passenger loadings on the network, Figure 10.9 shows similar but different outputs to the flow loadings for the deterministic models. First, the similar outputs mean that the stochastic model creates similar outputs because there is only a relatively small effect of the entropy term over the generalized travel cost from origin to destination. Second, the differences among the proposed stochastic models are generated by several reasons: differences in the final (floating number) flows, and differences in the capacity and entropy cost affected by the final flows. On the other hand, the deterministic result in Figure 10.6 shows more consistent results across the various algorithms than the results of the stochastic models because the deterministic models do not consider the entropy terms on the generalized cost.

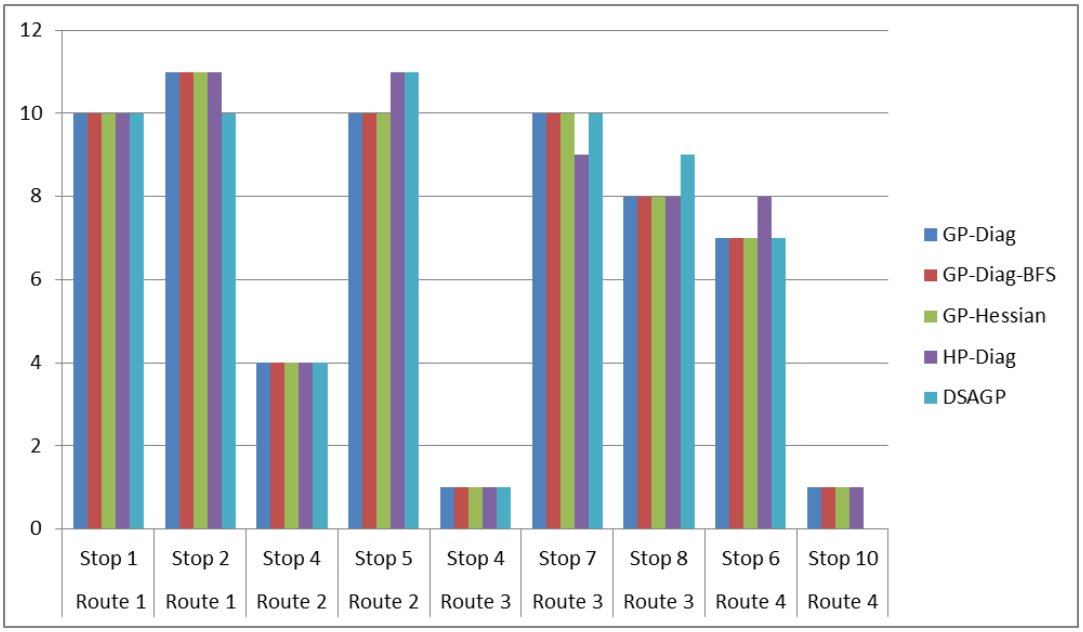


Figure 10.9 Passenger Loadings of the Proposed Stochastic Models

10.3 Real Network Test

10.3.1 Test Area

For the real network test of the proposed models, we prepared a partial area of the Sacramento region, including Downtown, East Sacramento, and Rancho Cordova as shown in Figure 10.10. This area is the southern area of regional Sacramento, geographically separated by the American River passing through Sacramento.

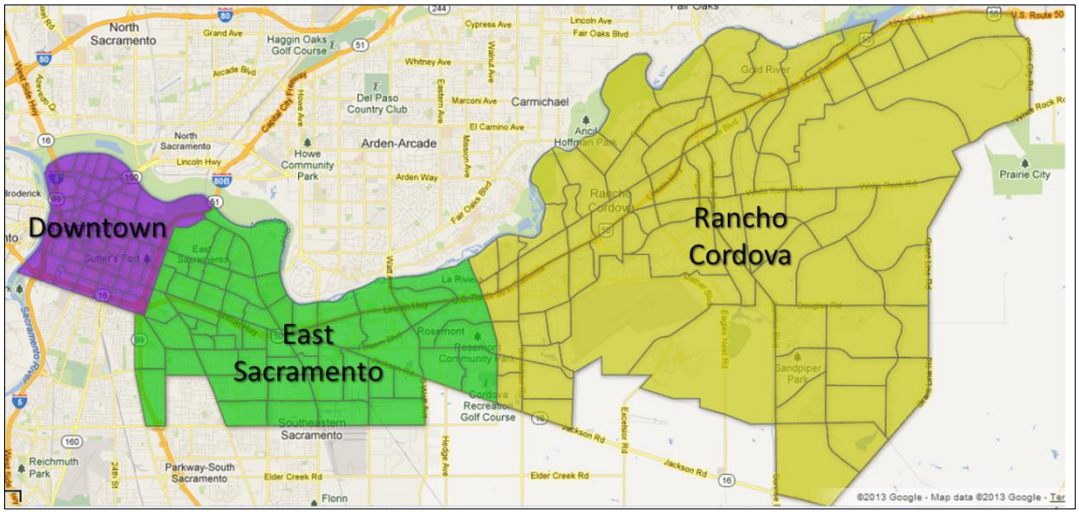


Figure 10.10 Test Area (Downtown - East Sacramento - Rancho Cordova)

The transit service within the study area consists of 17 transit routes including two light rail lines, the "Gold line" and the "Blue line", and 15 bus routes as shown in Figure 10.11. These 17 routes over 1,065 stops are run by 885 transit vehicle trips with 31,564 scheduled stop-times. The main transit system connecting these three subareas is the "Gold line" light rail from Downtown to Folsom. In addition to the schedule links, 2,378 transfers are prepared within a 0.25-mile Euclidean radius of each stop.

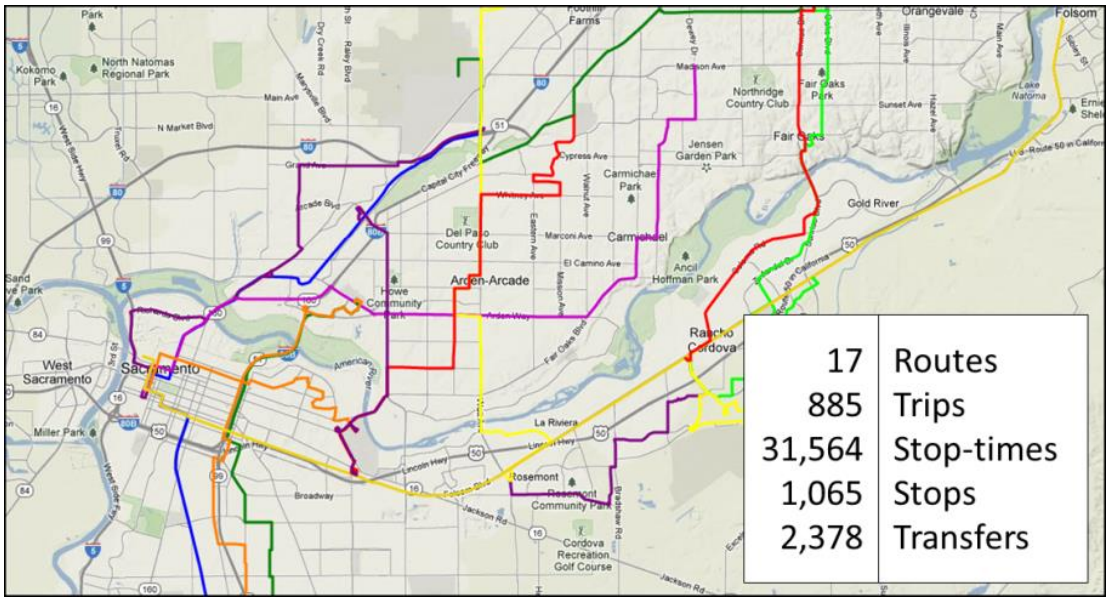


Figure 10.11 Transit Network of Test Area

For the test, 13,902 passengers were taken from the Sacramento regional tour-based activity-based model (ABM), called “DaySim”, and the daily passenger-trip pattern is shown in Figure 10.12. The demand shows typical AM and PM peak trips coinciding with the PAT and PDT. Most of the origin-destination (O-D) trips in the AM peak show a PAT-oriented pattern and the PM peak shows a more PDT-oriented pattern.

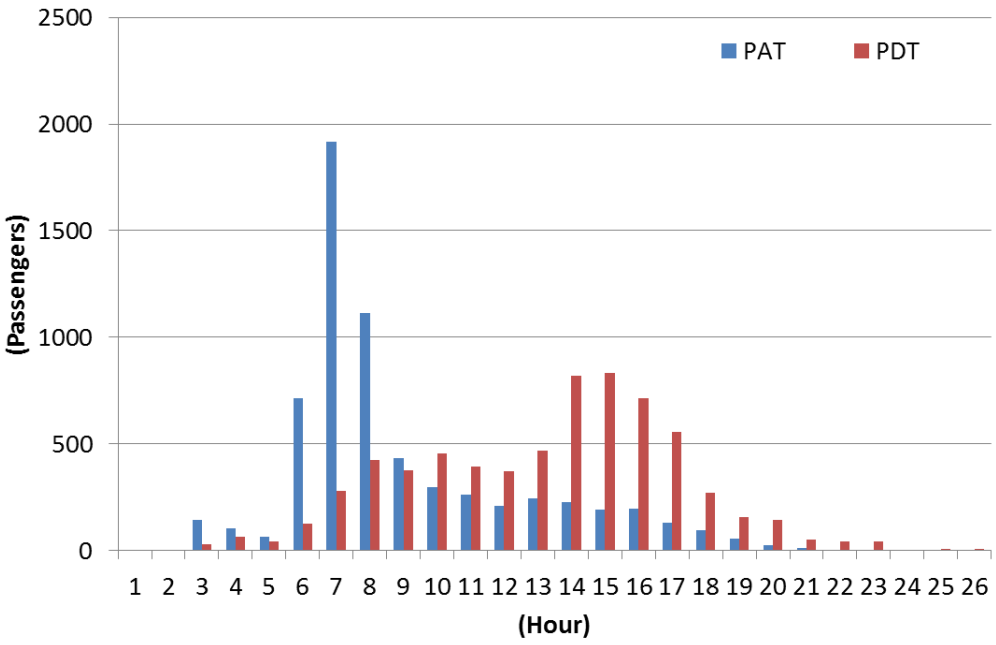
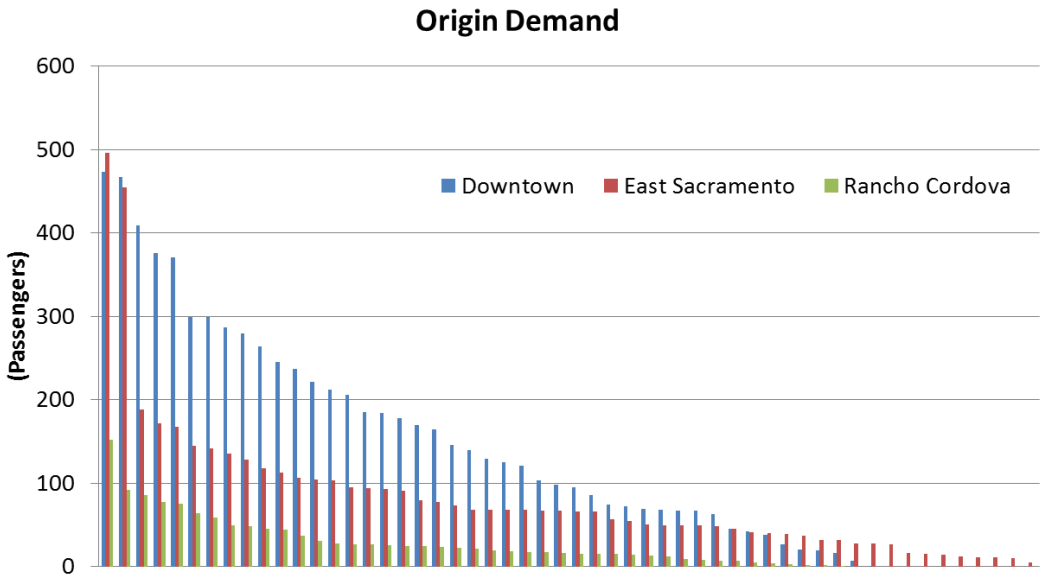
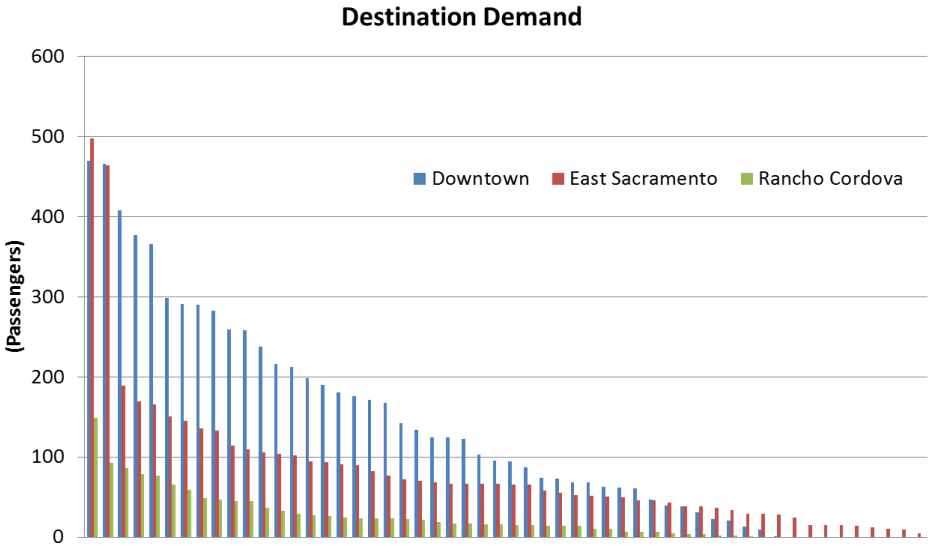


Figure 10.12 Daily Demand Pattern for Test Area

The O-D demand pattern by traffic analysis zone (TAZ) in each regional subarea (Downtown, East Sacramento, and Rancho Cordova) is shown in Figure 10.13. The O-D demand in this figure is ordered by the number of passengers without considering the TAZ. The Downtown O-D demands among these subareas are higher than the other two areas. East Sacramento shows a moderate demand and Rancho Cordova shows the least demand among the three areas.



(a) Origin Demand



(b) Destination Demand

Figure 10.13 Origin and Destination Demand by Traffic Analysis Zone (TAZ)

The trip distribution from Subarea-to-Subarea is shown in Table 10.2. Around 65% of trips (8,523 trips) are assigned to intra-subarea trips (Downtown: 5,235, East Sacramento: 2,588, and Rancho Cordova: 700)

out of the total of 13,092 trips. O-D trips between Downtown and East Sacramento (1,686 and 1,670) are higher than the O-D trips with respect to Rancho Cordova (347 and 340 from/to Downtown and 260 and 266 from/to East Sacramento).

Table 10.2 Demand Distribution by Subarea-to-Subarea

O\D trips	Downtown	East Sacramento	Rancho Cordova	Sum
Downtown	5,235	1,686	347	7,268
East Sacramento	1,670	2,588	260	4,518
Rancho Cordova	340	266	700	1,306
Sum	7,245	4,540	1,307	13,092

10.3.2 Test Models

For testing the proposed models, above all, the parameters and related model configuration values are set as shown in Figure 10.14. However, for each model, we apply different values of "Max outer iteration for RelGap" and "Max inner iteration for RelGap". Definitions of the parameters and configuration values are mentioned in Chapter 9.3. For the detailed application, we investigated two main tests. One is a capacity reduction to examine how passengers respond to the capacity constraint, and the other is the test with full demand. We note that the calibration is not considered in this study.

1. Transit Assignment Configuration
Iteration (1) or relative gap (2): 2
Outer iteration number: 4
Inner iteration number: 10
Relative gap for inner loop: 0.00001
Relative gap for outer loop: 0.001
Max outer iteration for RelGap: 100
Max inner iteration for RelGap: 100

2. Time-expanded Network Configuration
Max time prism from departure time (min): 200
Access and egress distance to stop (mile): 1.0
Number of max access nodes: 8

3. Hyperpath Search Configuration
PAT time boundary (min): 30
PDT time boundary (min): 30
Time Interval Bin Resolution (min): 5

4. Hyperpath Search Parameters
Number of transfers: 15.0
Travel time: 1.0
Transfer time: 3.0
Waiting time: 3.0
Access time: 1.0
Egress time: 1.0
Early arrival for PAT: 1.2
Late departure for PDT: 1.2
Logit sensitivity (Theta): -1.0
Alternative acceptance boundary in hyperpath: 0.01
Probability accepted for alternative: 0.001

5. Path-Overlapping Control Parameters
Path-size logit(PSL) beta in logit: 1.0
Path-size logit(PSL) gamma in PS: 2.0

6. Gradient Projection Parameters
Flow-shift alpha: 1.0

7. Capacity Cost Parameter
Capacity Alpha: 3.0
Capacity Cost Model(exponential: 1, power: 2): 1

8. Disaggregate Self-Adaptive Gradient Projection
Initial alpha[>0]: 10
Parameter u in alpha[0.5<=u<=1]: 0.8
Parameter delta in Condition 1[0<delta<1]: 0.5
Max l-parameter: 200
Max alpha-parameter: 9999
Epsilon for convergence: 0.001

Figure 10.14 Parameters and Configuration Values (*input_ft_parameters.dat*)

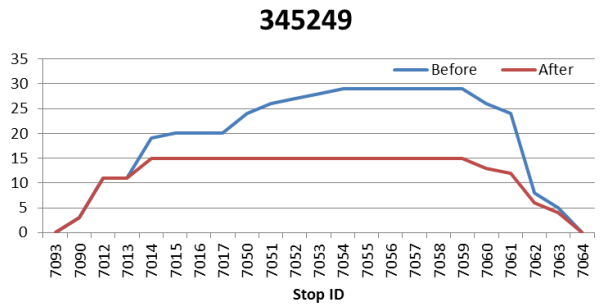
10.3.3 Test 1: Capacity Reduction

10.3.3.1 Demand and Test Preparation

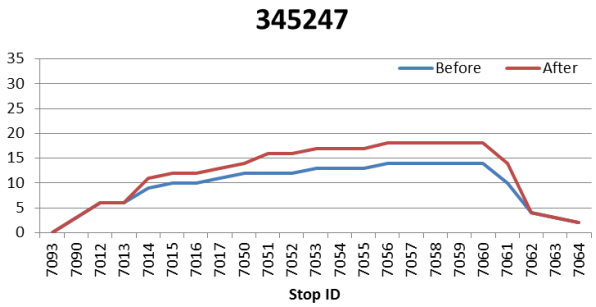
To determine the passenger movements according to limited capacities, we apply a test using a capacity reduction for specific routes. First, trip 345249, which runs along the light rail transit (LRT) Gold line from Downtown to Rancho Cordova, is considered for a single trip capacity reduction. The trip 345249 departs Downtown at 16:43 and the original capacity (300 passengers) of each LRT vehicle is artificially reduced to 15 passengers. Second, we consider multiple capacity reductions by adding one more capacity reduction to 17 passengers for Gold line trip 345248 departing at 16:58 after trip 345249 with a capacity of 15 passengers.

10.3.3.2 Test Results

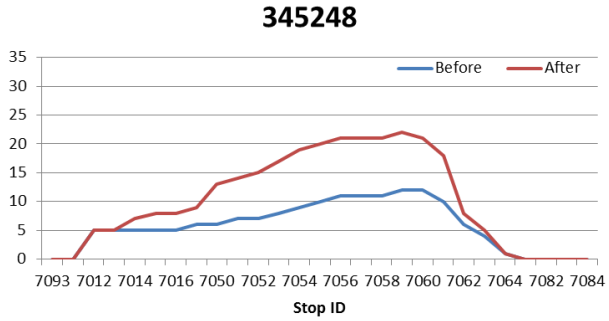
The capacity reduction is applied on trip 345249, and the overflows are shifted to other trips, primarily trip 345247 and 345248 as shown in Figure 10.15, in which trip 345247 and 345248 depart at 17:13 and 16:58, respectively. When we consider the afternoon trips from Downtown to Rancho Cordova, a pattern of shifting flows to later trips according to the PDT is expected.



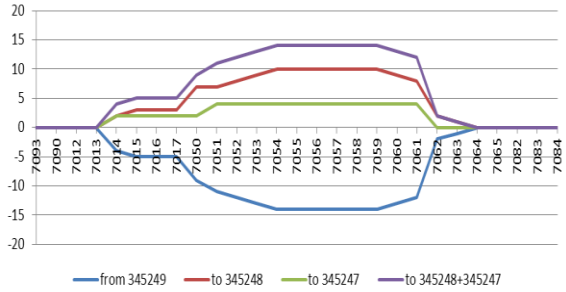
(a) Capacity Reduction (15 Passengers)



(b) Flow Increase on Trip 345247



(c) Flow Increase on Trip 345248



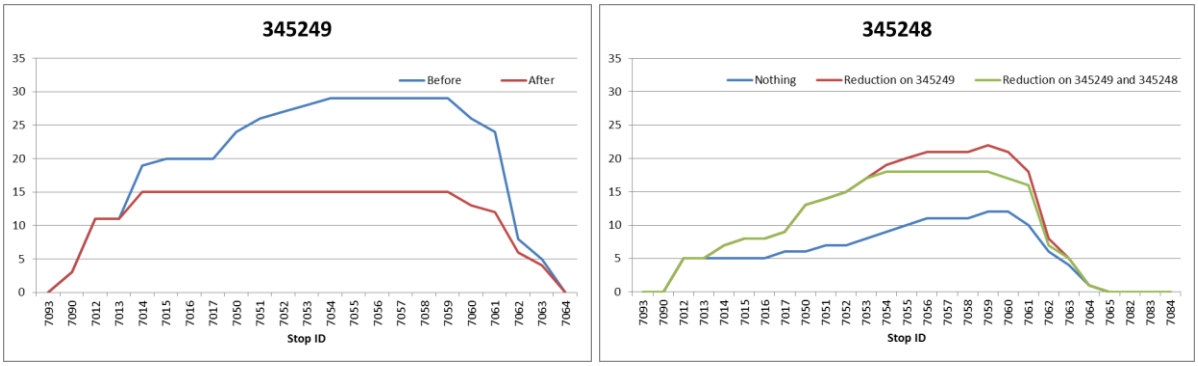
(d) Flow Shift by Capacity Reduction

Figure 10.15 Capacity Reduction on Trip 345249 (15 passengers)

The flows shifted off of trip 345249 are shifted to trips 345247 and 345248 as shown in Figure 10.15. Especially Figure 10.15(d) shows that all shifted flows on trip 345249 are the same as those flows shifted to trips 345247 and 345248 which show the flows affected by the capacity are thoroughly distributed to other available (later) transit trips.

When we have the additional capacity reduction on trip 345248 to 17 passengers, the result of the flow shift is presented in Figure 10.16. Figure 10.166(a) shows the flow reduction on trip 345249 which is similar to the example with a single capacity reduction, and Figure 10.16(b) shows more detail on the flow shifted from/to trip 345248. “Nothing” stands for no capacity reduction on both trips 345248 and

345249. “Reduction on 345249” means the only capacity reduction on trip 345249, which is the same as the previous example depicted on Figure 10.15(c). Finally, “Reduction on 345249 and 345248” shows the capacity reductions on both trips, which pushes more flows to other trips by retaining the capacity constraint.



(a) Capacity Reduction (15 Passengers) (b) Flow Shift from Trip 345248

Figure 10.16 Capacity Reduction on Trip 345249 and 345248

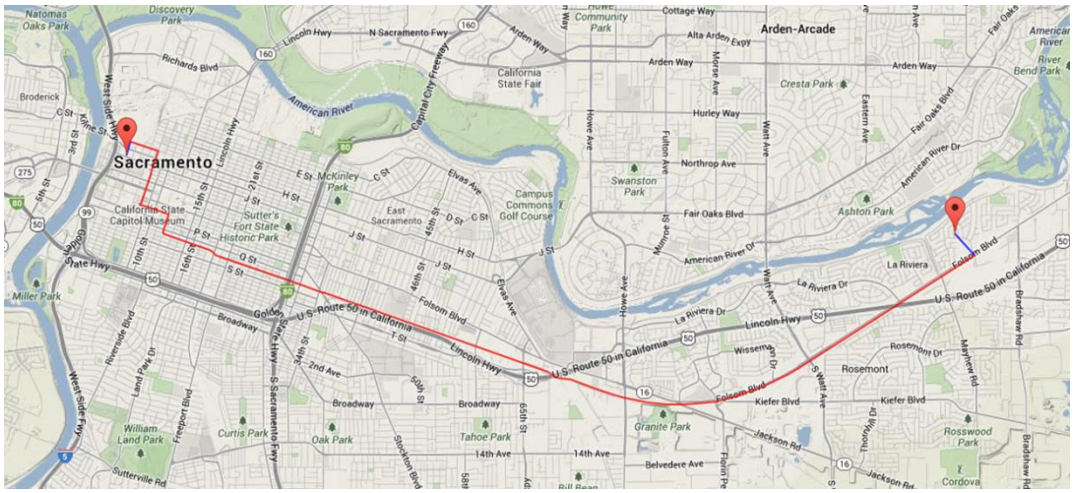
10.3.4 Test 2: Test over the Study Area

This test includes the overall test of the proposed models over the study area. For the test set-up, we follow the parameters given in Figure 10.14. We also note that the study area does not have transit services running over the entire area, which is intentionally assumed to create a capacitated transit system with the passenger demand.

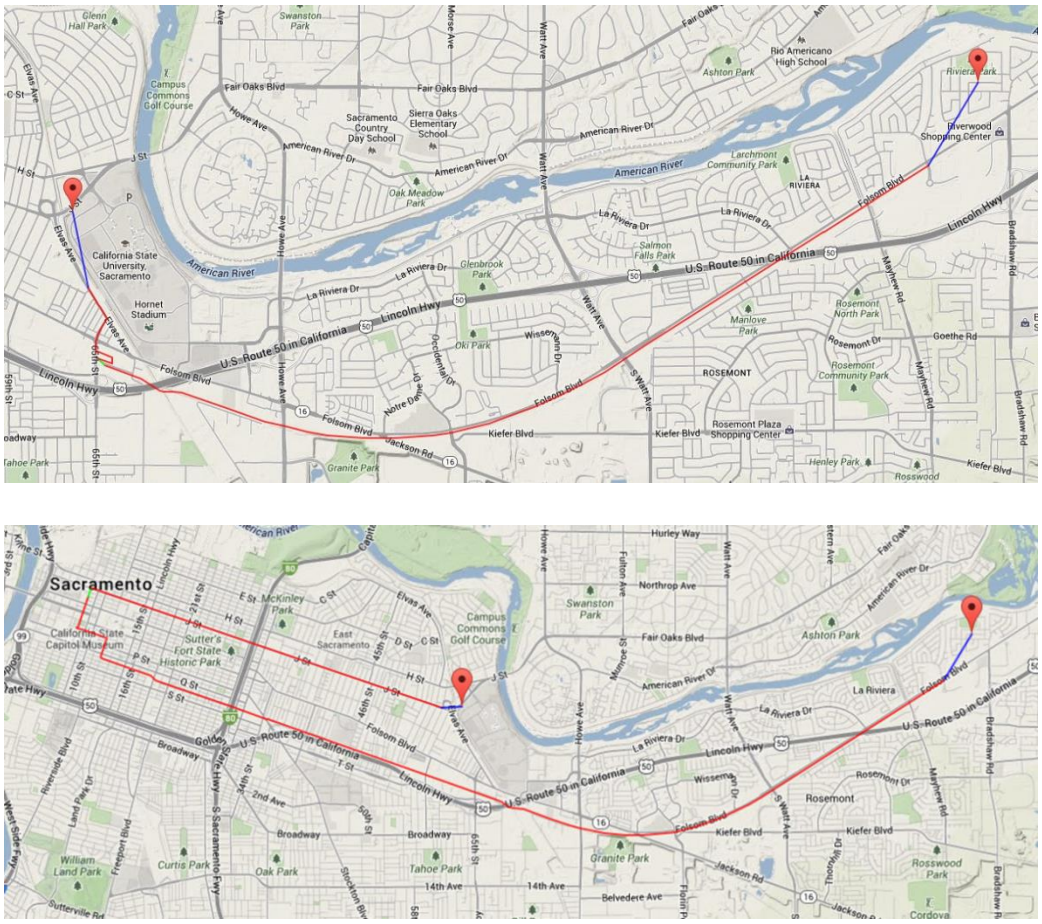
10.3.4.1 Path Enumeration

For the path enumeration, we use the proposed hyperpath model in this study, considering different parameters. In this case, we applied the same generalized (or disutility) cost except for early arrival or late departure cost which is decreased to 1/10 to have, at least, multiple alternatives on the same transit route.

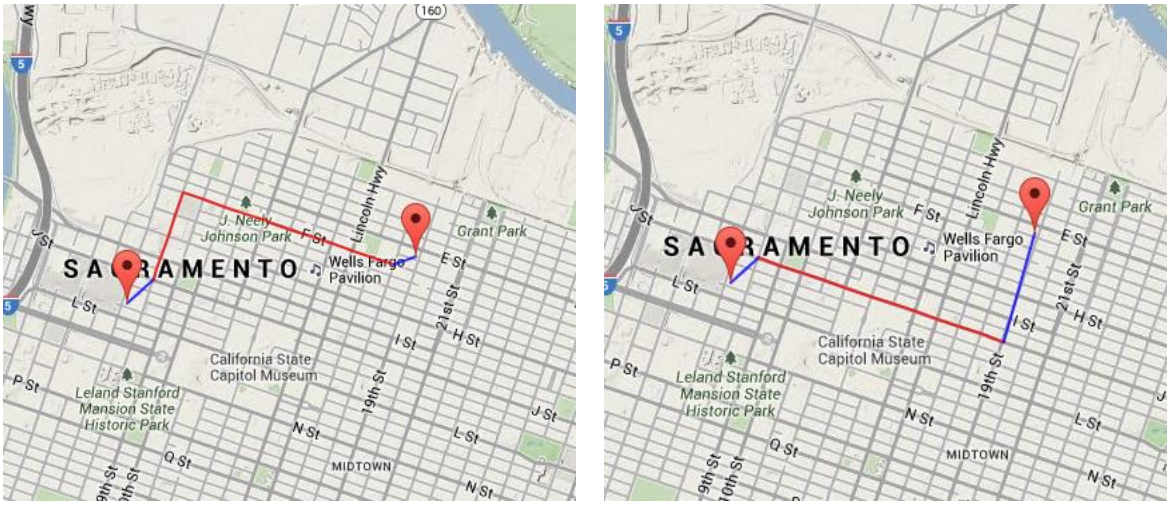
Above all, alternative paths from Rancho Cordova (TAZ 599) to Downtown (TAZ 766) are shown in Figure 10.17 (a) which has 8 different trips on the same route, the Gold line. For Rancho Cordova (TAZ 591) to East Sacramento (TAZ 514), the alternative paths are shown in Figure 10.17 (b) by two different paths, which include multiple trips on each path, including transfers. For Downtown (TAZ 806) to Downtown (TAZ 802), multiple trips are detected along the two different paths in Figure 10.17 (c).



(a) Rancho Cordova (TAZ 599) to Downtown (TAZ 766)



(b) Rancho Cordova (TAZ 591) to East Sacramento (TAZ 514)



(c) Downtown (TAZ 806) to Downtown (TAZ 802)

Figure 10.17 Path Enumeration Examples

10.3.4.2 Deterministic Models

The convergence of the proposed deterministic models is shown in Figure 10.18. The left and right of the figure show computational times of 0 to 1,600 seconds for GP models and from 1601 to 73,000 seconds for DSAGP. Above all, GP-Diag and GP-Diag-BIS show the same pattern of convergence, with relatively big oscillations according to frequent diagonalization iterations until the model converges. We also notice that GP-Diag has better performance than GP-Diag-BIS. DSAGP shows very long computationation time, which is represented by the relative gap of the VI objective value, taking around 72,000 seconds (1,200 minutes) by hitting the predefined maximum of 20 iterations. GP-Hessian also takes a relatively longer computation time than the other GP models.

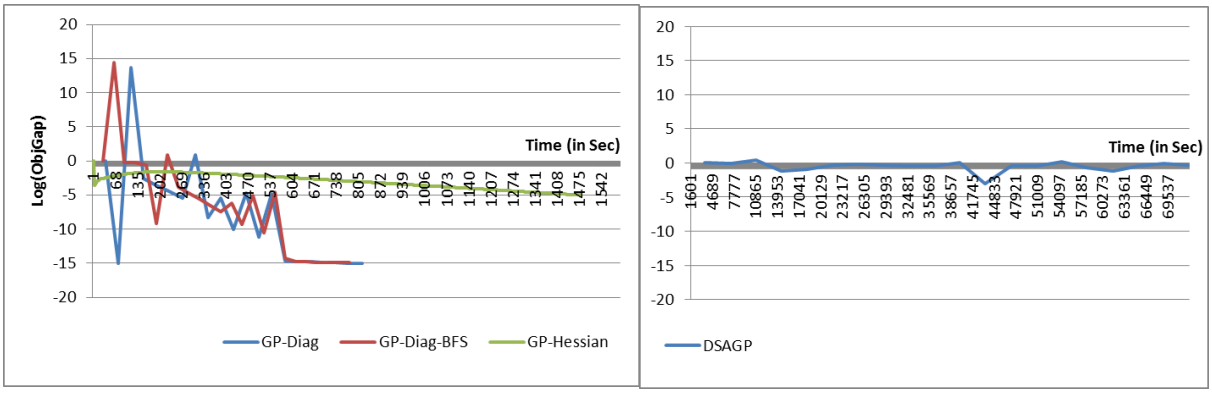


Figure 10.18 Convergence of Deterministic Models

For the performance of the proposed deterministic models, Figure 10.19 shows noticeable differences between DSAGP and other models (GP-Diag, GP-Diag-BIS, and GP-Hessian).

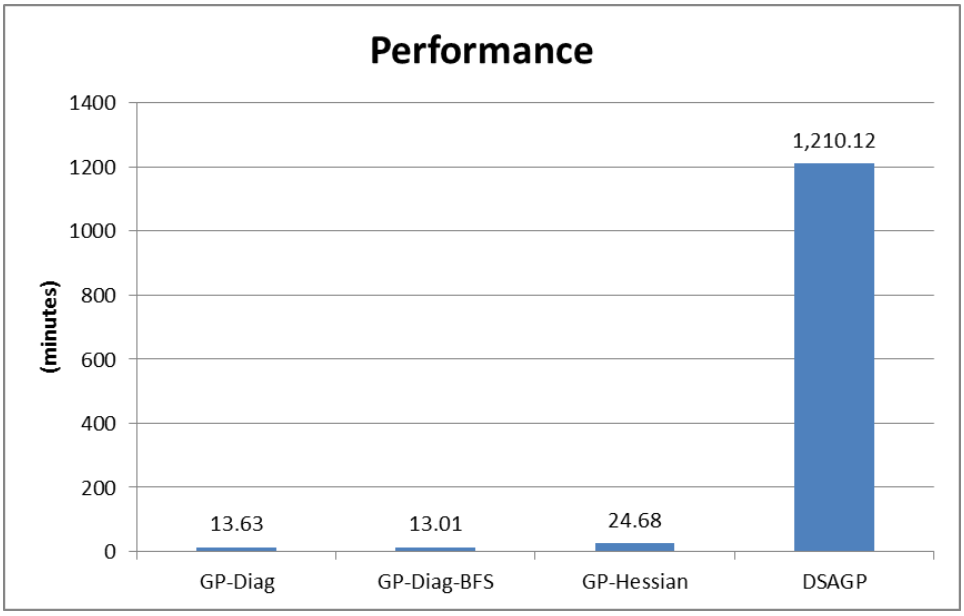


Figure 10.19 Computation Performances of Deterministic Models

Different from the simple network test in Figure 10.5, DSAGP requires significant computation time. As discussed earlier, this is because the inner loop of estimating appropriate step-size α requires a long process for each O-D pair. Also, in Figure 10.19, GP-Diag-BIS shows a little better performance than GP-Diag, although it takes more computation time as shown in Figure 10.18.

For the outputs of the passenger assignments, overall statistics are shown in Figure 10.20. Among the proposed deterministic models, the results are almost the same. Overall O-D travel distances are around 3.2 miles and total travel times are around 24.2 minutes. The result also shows a relatively small number of transfers, around 0.24, and transfer time per passenger, around 0.74 minutes. Access and egress walking times are estimated to be 5.73 and 5.91 minutes. Stop-to-stop "travel time" in Figure 10.20, which is defined by stop-to-stop travel time excluding access and egress walking times, averages 12.54 minutes. In conclusion, these results indicate that the transit service of the study area provides relatively short trips using both single transit trips as well as providing appropriate transit accessibility.

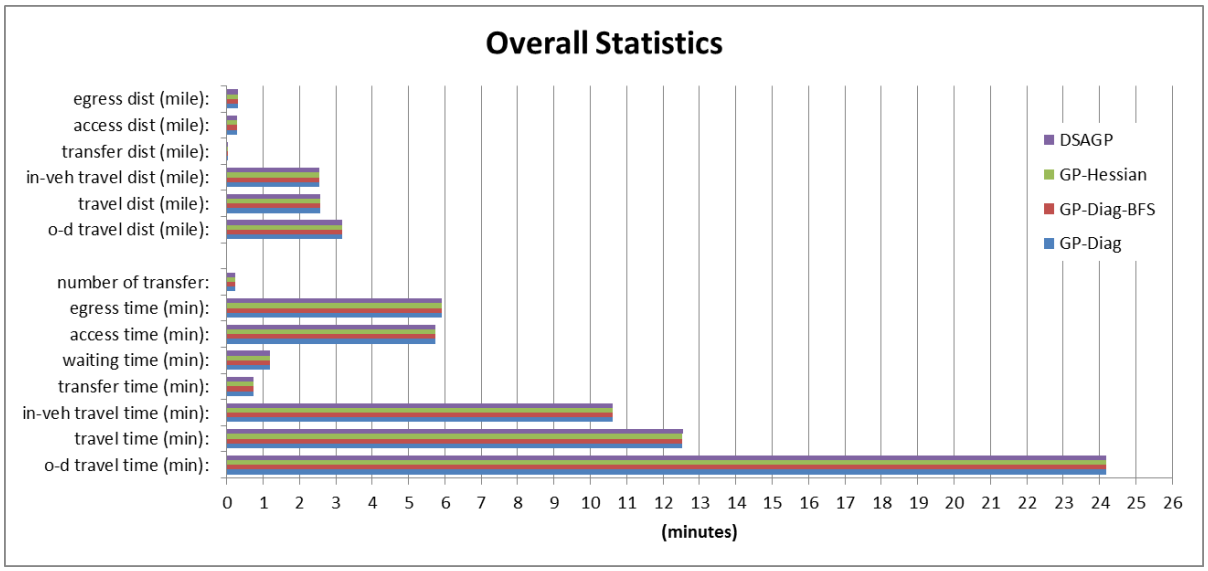


Figure 10.20 Overall Statistics for Deterministic Models

For a detailed comparison among models, we compared whole individual transit passenger loadings based on GP-Hessian, which is the number of loaded passengers for each stop of each transit trip shown in Figure 10.21. The 45 degree shows the exact match between the other proposed models and GP-Hessian. The dots located in higher than 45 degree shows more passenger loadings than the passenger loadings of GP-Hessian and vice versa. GP models (GP-Diag, GP-Diag-BIS, GP-Hessian) show almost the same result following the 45 degree alignment, meaning that GP models produce a similar result. On the other hand, DSAGP shows a different solution plot, although in the overall statistics is almost the same as in Figure 10.20. Especially over the area around 50 ~60 flows (on X-axis), we see several significant drops.

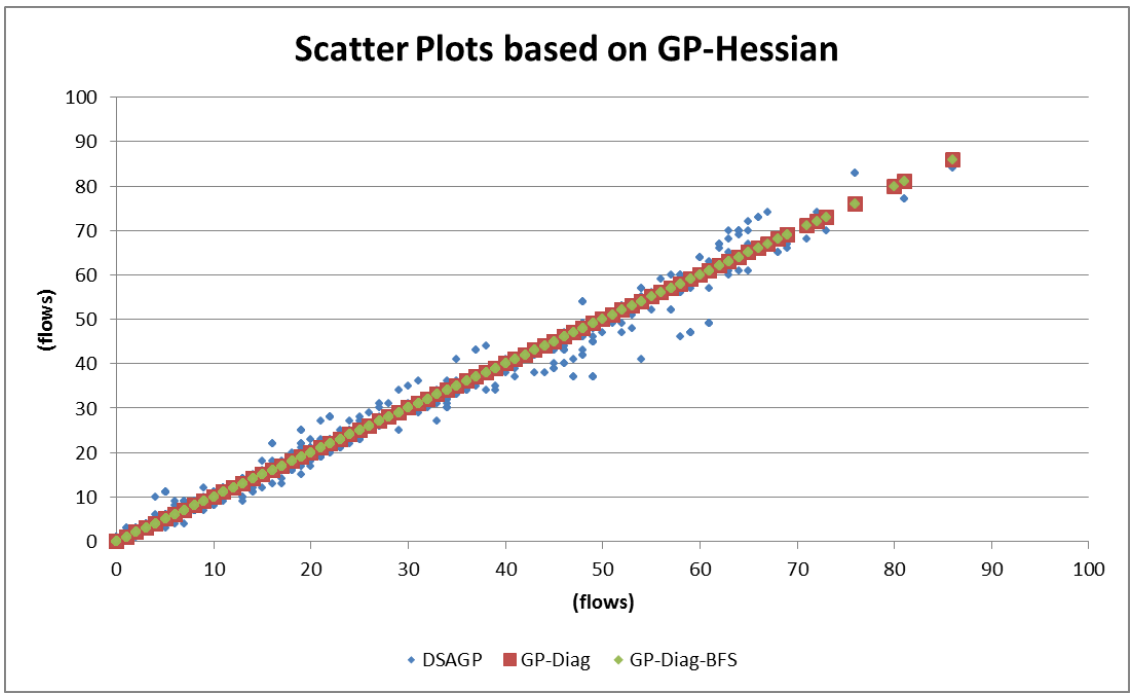


Figure 10.21 Scatter Plots of Deterministic Models Based on GP-Hessian

10.3.4.3 Stochastic Models

For the analysis of the stochastic models, we have added HP-Diag models using the hyperpath-based MSA model. The convergence result of the proposed models is shown in Figure 10.22. The left and right figure show from 0 to 1600 seconds for GP models and from 1601 to 73,000 seconds for HP-Diag and DSAGP models, respectively. Except for HP-Diag, the stochastic models show a similar result to the result of the deterministic models. This allows us to conjecture that the entropy term of the proposed stochastic models is not effectively large enough to push the flows to other available paths. As mentioned, GP-Diag and GP-Diag-BIS show strong oscillation within a small number of iterations and, on the other hand, GP-Hessian plots a gradual convergence rate without noticeable peaks like GP-Diag or GP-Diag-BIS. DSAGP also shows a very long computation time, around 72,000 seconds as well as reaching the maximum 20 iterations, and HP-Diag demonstrates gradual oscillations until the model converges, which also took a relatively long computation time, around 52,700 seconds.

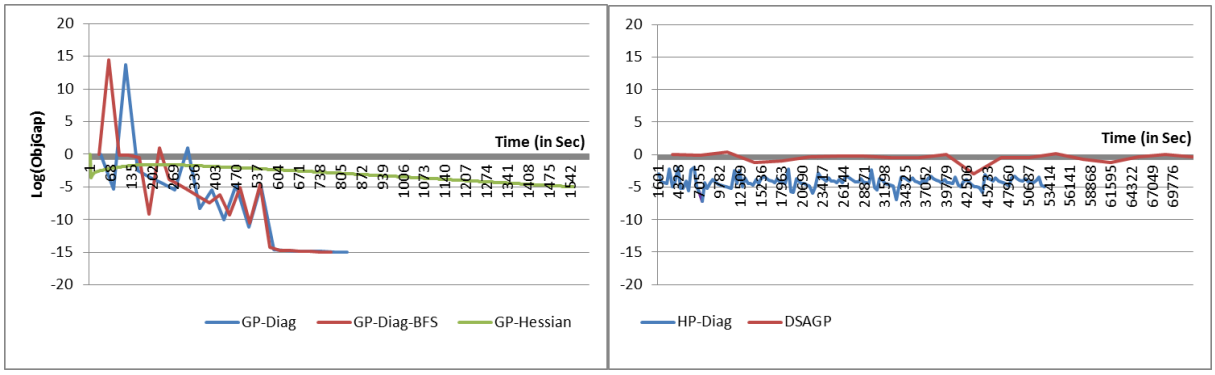


Figure 10.22 Convergence of Stochastic Models

The performance of the model in Figure 10.23 shows a similar pattern with subtle changes and improved HP-Diag computational performance. The computation times of GP-Diag and GP-Hessian are increased a little to 13.78 and 25.85 minutes (16.36 and 24.68 minutes of the results of deterministic models shown in Figure 10.19). On the other hand, DSAGP improves its computational time from 1,210.12 to 1,153.45 minutes and GP-Diag-BIS also improves a little around 0.11 minutes from 13.01 to 12.90 minutes. HP-Diag model shows slow performance because of the MSA models using a hyperpath search in every iteration.

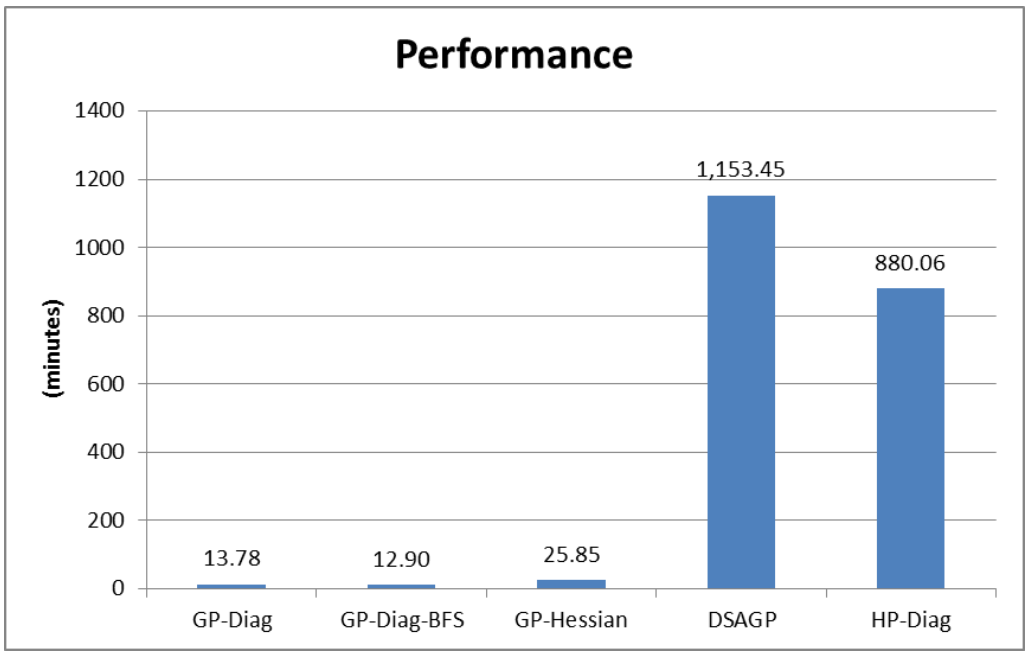


Figure 10.23 Computational Performance of Stochastic Models

Figure 10.24 shows the overall statistics of the stochastic transit assignment. As discussed earlier, the results show similar outputs to the outputs of the deterministic models, in terms of convergence. However, HP-Diag represents a little different statistics. Total O-D travel time is estimated to be 24.53 minutes longer than the estimated 24.18 minutes of other models. We can notice easily the reason in Figure 10.24, in that access and egress walking times are estimated to bearound 0.18 minutes longer than the other models. The main difference of HP-Diag is in utilizing hyperpath search in every iteration, which examines the fastest hyperpath not only including transit trip links but also searching new access and egress links. So the HP-Diag model has more possibilities to have different access and egress links than other GP and DSAGP models using predefined access and egress links. Excluding these access and egress travel times, HP-Diag estimates a similar “travel time”, 12.52 minutes against 12.54 from other GP and DSAGP models.

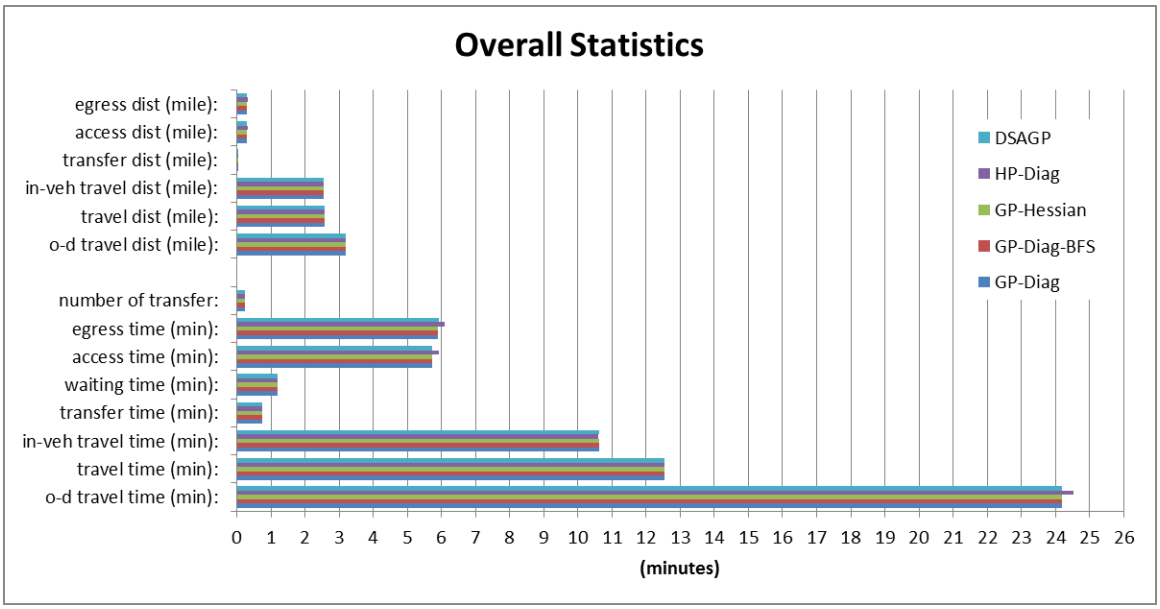


Figure 10.24 Overall Statistics of Stochastic Models

To scrutinize the similarity of the passenger loading outputs, we present the scatter plots for the proposed stochastic models in Figure 10.25. As we depict for the deterministic models in Figure 10.21, the scatter plot is based on GP-Hessian and loading along 45 degree represents the exact match to GP-Hessian model outputs. First, except for the loading flows from 0 to 20 passengers and around 30 passengers on the X-axis, GP-Diag and GP-Diag-BIS almost follows the 45 degree of GP-Hessian models. Second, HP-Diag models are scattered along the 45 degree line of GP-Hessian and most of the scattered loadings are located from 0 to 40 units of flow on the X-axis. Finally, DSAGP shows a similar pattern to the result from the deterministic DSAGP.

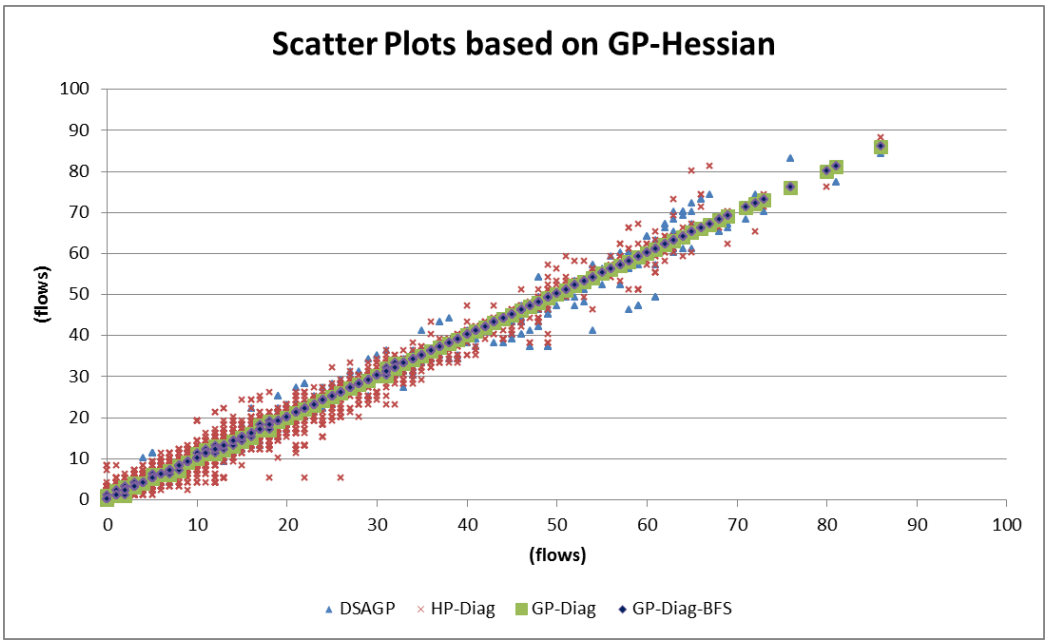


Figure 10.25 Scatter Plots of Stochastic Models Based on GP-Hessian

11 CONCLUSION

The main contribution of this study is in the development of schedule-based transit assignment models, especially achieving a stochastic passenger behavior on a congested transit network as well as a deterministic passenger behavior under congestion. For this contribution, we proposed several transit assignment models including path- and hyperpath-based models and a self-adaptive model considering deterministic and stochastic passenger behaviors. *First*, a hyperpath-based transit assignment model was implemented by the method of successive averages (MSA) in which the hyperpath search was executed by the link-based hyperpath (LBHP) algorithm for stochastic path generation. *Second*, for better performance, we proposed two path-based transit assignment models, using gradient projection including diagonal elements or a full Hessian matrix. *Third*, we also explored a possibility of DSAGP model using the self-adaptive technique proposed by Chen et al. (2012). In addition, we have also investigated the possibility of providing efficient performance, by initiating a feasible solution that is close to the deterministic UE solution, called the deterministic better initial solution (D-BIS). This initialization method was also applied to the stochastic assignment models, as it was assumed that the stochastic models would have similar results to the deterministic models. We also realized that a better initial solution on a stochastic model is hard to be estimated within the proposed transit assignment models. The proposed hyperpath- and path-based and DSAGP transit assignment models were tested on a simple test network and a real transit network from the Sacramento area.

In terms of model application, we recommend the GP-Diag-BIS model for a large network application, based on the computation performance. By applying BIS, it guarantees the advantage of good computation performance that the transit passenger behavior pattern on a congested transit system can be estimated in advance. When we consider BIS, deterministic-BIS (D-BIS) is also recommended for both deterministic and stochastic assignment models, since the entropy term on each path cost is relatively weak. This is because the transit passenger demand by O-D-T is small, especially with finer resolution of

time interval T, i.e., 5 minutes. We also realized that GP-Diag also showed solid computation performance without the pre-processing, like BIS, if the priority of boarding is relatively plain such that most of priority paths are close to the shortest path. And GP-Hessian is another appropriate model, although it requires longer iterations in the real application, since the path-based approach using enumeration allows a fast performance even with these long iterations.

For creating paths in the proposed assignment models, we have proposed various techniques, such as a label-setting shortest path, label-correcting and -setting hyperpath models, and hierarchical versions of those path search models, on a link-based time-expanded (LBTE) transit schedule network. The LBTE network more effectively captures turning movements like transfers easily as well as maintaining the efficient size of a schedule-based network. The proposed hyperpath model on a transit schedule network, which was introduced by Noh et al. (2012a), was mainly used for creating an alternative set of paths considering a logit-type passenger's behavior as well as directly employing it in hyperpath-based transit assignment.

To consider a congested transit system, a "soft-capacity" function is critically defined to represent a relation between each transit vehicle's capacity and the passengers' priority of boarding. This capacity cost is included as a transfer cost, which is especially appropriate to the proposed LBTE transit schedule network. The resulting asymmetric passenger cost relation, including the priority of boarding within a congested transit service, was considered properly and implemented appropriately over both the test network and on the real Sacramento case study.

During the development of the transit assignment models, we have realized that there are numerous considerations for future researches. *First*, we applied the hyperpath search model for enumerating paths, especially for path-based and DSAGP assignment models. In terms of creating a path set for a transit assignment, this model showed that adequate alternatives can be found by a hyperpath search on a transit schedule network. However, we may think about whether a set of alternatives in a hyperpath is sufficient

with respect to representing a passenger's behavior, and we may consider other path generation models like path generation using a *k*-shortest path search. *Second*, for the better performance of the proposed models, we have mainly explored path-based gradient projection models (GP-Diag, GP-Diag-BIS, and GP-Hessian) for deterministic and stochastic transit assignments. Considering that the proposed fundamental models are originated from auto-centered traffic assignment, it is worthwhile to study further cutting-edge assignment models like bush-based traffic assignment models, and to explore integration with the auto mode in an intermodal assignment, building from this proposed model foundation. *Third*, we need to consider calibrating these models for real transportation applications. We have developed several significant transit assignment models and did not aim to calibrate the proposed models. However, to represent a suitable passenger behavior, the various parameters shown in the model development should be calibrated, especially for parameters of the hyperpath (generalized cost), the PSL correction, the capacity cost, and the logit assignment models. *Fourth*, among the proposed models, HP-Diag and DSAGP showed relatively worse performance than other models in terms of computation time. To apply the models within an acceptable computation time, we may consider additional researches with the proposed models to improve the performance, such as parallel or multi-core computing, not only improving HP-Diag and DSAGP but also expediting GP-Diag, GP-Diag-BIS, and GP-Hessian models.

REFERENCES

Ahuja, R. K., T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, Englewood Cliffs, New Jersey, 1993.

Armijo L. Minimization of Functions Having Continuous Partial Derivatives. *Pacific Journal of Mathematics*, Vol.16, No.3, 1966, pp.1–3.

Bekhor, S., and T. Toledo. Investigating Path-Based Solution Algorithms to the Stochastic User Equilibrium Problem. *Transportation Research Part B: Methodological*, Vol. 39, No. 3, 2005, pp. 279-295.

Ben-Akiva, M. E., and M. Bierlaire. Discrete Choice Methods and Their Applications to Short-Term Travel Decisions. In Handbook of Transportation Science (R. W. Hall, ed.), Kluwer Academic Publishers, Boston, Mass., 1999, pp. 5–33.

Bertsekas, D. On the Goldstein-Levitin-Polyak Gradient Projection Method. *Automatic Control, IEEE Transactions*, Vol. 21, No. 2, 1976, pp. 174-184.

Bertsekas, D. Projected Newton Methods for Optimization Problems with Simple Constraints. *SIAM J. Control and Optimization*, Vol. 20, No. 2, 1982, pp. 221-246.

Carraresi, P., F. Malucelli, and S. Pallottino. Regional Mass Transit Assignment with Resource Constraints. *Transportation Research Part B: Methodological*, Vol. 30, No. 2, 1996, pp. 81-98.

Cascetta, E., A. Nuzzolo, F. Russo, and A. Vitetta. A Modified Logit Route Choice Model Overcoming Path Overlapping Problems: Specification and Some Calibration Results for Interurban Networks. In *Transportation and Traffic Theory: Proc., 13th International Symposium on Transportation and Traffic Theory* (J. B. Lesort, ed.), Pergamon, Burlington, Mass., 1996.

Cepeda, M., R. Cominetti, and M. Florian. A Frequency-Based Assignment Model for Congested Transit Networks with Strict Capacity Constraints: Characterization and Computation of Equilibria. *Transportation Research Part B: Methodological*, Vol. 40, No. 6, 2006, pp. 437-459.

Chen, H. -. *Dynamic Travel Choice Model: A Variational Inequality Approach.* Springer, 1999.

Chen, A., and D. Lee. Path-Based Algorithms for Large-Scale Traffic Equilibrium Problem: A Comparison between DSD and GP In: Proceedings of the 78th Annual Meeting of Transportation Record Board, 1999.

Chen, A., D. Lee, and R. Jayakrishnan. Computational Study of State-of-the-Art Path-Based Traffic Assignment Algorithms. Mathematics and Computers in Simulation, Vol. 59, No. 6, 2002, pp. 509-518.

Chen A., Z. Zhou, X. Xu. A Self-adaptive Gradient Projection Algorithm for the Nonadditive Traffic Equilibrium Problem, *Computers &Operations Research*, Vol. 39, 2012, pp. 127–138.

Chriqui, C., and P. Robillard. Common Bus Lines. *Transportation Science*, Vol. 9, No. 2, 1975, pp. 115-121.

Cominetti, R., and J. Correa. Common-Lines and Passenger Assignment in Congested Transit Networks. *Transportation Science*, Vol. 35, No. 3, 2001, pp. 250-267.

Daganzo, C. F., and Y. Sheffi. On Stochastic Models of Traffic Assignment. *Transportation Science*, Vol. 11, No. 3, 1977, pp. 253-274.

De Cea, J., and E. Fernandez. Transit Assignment for Congested Public Transport Systems: An Equilibrium Model. *Transportation Science*, Vol. 27, No. 2, 1993, pp. 133-147.

Dial, R. B., A Probabilistic Multipath Traffic Assignment Model which Obviates Path Enumeration. *Transportation Research*, Vol. 5, No. 2, 1971, pp. 83-111.

Dial, R. B., A. M. Voorhees, and Associates Inc. Transit Pathfinder Algorithm. *Highway Research Record*, No. 205, 1967, pp. 67-85.

Dijkstra, E. W. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, Vol. 1, No. 1, 1959, pp. 269-271.

Evans, S. P. A Relationship between the Gravity Model for Trip Distribution and the Transportation Problem in Linear Programming. *Transportation Research*, Vol. 7, No. 1, 1973, pp. 39-61.

Fisk, C. Some Developments in Equilibrium Traffic Assignment. *Transportation Research Part B: Methodological*, Vol. 14, No. 3, 1980, pp. 243-255.

Gallo, G., G. Longo, S. Pallottino, and S. Nguyen. Directed Hypergraphs and Applications. *Discrete Applied Mathematics*, Vol. 42, No. 2-3, 1993, pp. 177-201.

Hamdouch, Y., and S. Lawphongpanich. Congestion Pricing for Schedule-Based Transit Networks. *Transportation Science*, Vol. 44, No. 3, 2010, pp. 350-366.

———. Schedule-Based Transit Assignment Model with Travel Strategies and Capacity Constraints. *Transportation Research Part B: Methodological*, Vol. 42, No. 7-8, 2008, pp. 663-684.

Hamdouch, Y., P. Marcotte, and S. Nguyen. Capacitated Transit Assignment with Loading Priorities. *Mathematical Programming*, Vol. 101, No. 1, 2004, pp. 205-230.

———. A Strategic Model for Dynamic Traffic Assignment. *Networks and Spatial Economics*, Vol. 4, No. 3, 2004, pp. 291-315.

He, B. S., H. Yang, Q. Meng, D. R. Han. Modified Goldstein–Levitin–Polyak Projection Method for Asymmetric Strongly Monotone Variational Inequalities, *Journal of Optimization, Theory and Applications*, Vol. 112 No. 1, 2002, pp. 129–43.

Hoogendoorn-Lanser, S., R. van Nes, and P. Bovy. Path Size Modeling in Multimodal Route Choice Analysis. *Transportation Research Record: Journal of the Transportation Research Board*, No. 1921, Transportation Research Board of the National Academies, Washington, D.C., 2005, pp. 27–34.

Hoogendoorn-Lanser, S., and P. Bovy. Modeling Overlap in Multimodal Route Choice by Including Trip Part–Specific Path Size Factors. In Transportation Research Record: Journal of the Transportation Research Board, No. 2003, Transportation Research Board of the National Academies, Washington, D.C., 2007, pp. 74–83.

Jayakrishnan, R., W. K. Tsai, J. N. Prashker, and S. Rajadhyaksha. Faster Path-Based Algorithm for Traffic Assignment. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 1443, 1994, pp. 75-83.

Khani, A., S. Lee, M. D. Hickman, H. Noh, and N. Nassir. Intermodal Shortest- and Optimal-Path Algorithm using a Transit Trip-Based Shortest Path. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2284, 2012a, pp. 40-46.

Lam, W. H. K., J. Zhou, and Z. Sheng. A Capacity Restraint Transit Assignment with Elastic Line Frequency. Transportation Research Part B: Methodological, Vol. 36, No. 10, 2002, pp. 919-938.

Lam, W. H. K., Z. Y. Gao, K. S. Chan, and H. Yang. A Stochastic User Equilibrium Assignment Model for Congested Transit Networks. *Transportation Research Part B: Methodological*, Vol. 33, No. 5, 1999, pp. 351-368.

Larsson, T., and M. Patriksson. Simplicial Decomposition with Disaggregated Representation for the Traffic Assignment Problem. *Transportation Science*, 26, 1992, pp. 4-17.

Le Clercq, F. A Public Transport Assignment Model. *Traffic Engineering and Control*, 1972, pp. 91–96.

Marcotte, P., S. Nguyen, and A. Schoeb. A Strategic Flow Model of Traffic Assignment in Static Capacitated Networks. *Operations Research*, Vol. 52, No. 2, 2004, pp. 191-212.

Nguyen, S., and S. Pallottino. Hyperpaths and Shortest Hyperpaths. In *Lectures Given at the Third Session of the Centro Internazionale Matematico Estivo (C.I.M.E.) on Combinatorial Optimization*, Springer-Verlag New York, Inc, New York, NY, USA, 1989, pp. 258-271.

———. Equilibrium Traffic Assignment for Large Scale Transit Networks. *European Journal of Operational Research*, Vol. 37, No. 2, 1988, pp. 176-186.

Nguyen, S., S. Pallottino, and F. Malucelli. A Modeling Framework for Passenger Assignment on a Transport Network with Timetables. *Transportation Science*, Vol. 35, No. 3, 2001, pp. 238-249.

Nielsen, L. R. *A Bicriterion and Parametric Analysis of the Shortest Hyperpath Problem*. Ph.D. Progress Report, Department of Operations Research, University of Aarhus, 2001.

Nielsen, L. R., K. A. Andersen, and D. Pretolani. Finding the K Shortest Hyperpaths. *Computers & Operations Research*, Vol. 32, No. 6, 2005, pp. 1477-1497.

Nielsen, O. A. A Stochastic Transit Assignment Model Considering Differences in Passengers Utility Functions. *Transportation Research Part B: Methodological*, Vol. 34, No. 5, 2000, pp. 377-402.

Nielsen, O. A., and R. D. Frederiksen. Large-Scale Schedule-Based Transit Assignment - further Optimization of the Solution Algorithms. In Operations Research/Computer Science Interfaces Series, Vol. 46, Kluwer Academic Publishers, Boston, Mass., 2009, pp. 1–26.

———. Optimisation of Timetable-Based, Stochastic Transit Assignment Models Based on MSA. *Annals of Operations Research*, Vol. 144, No. 1, 2006, pp. 263-285.

Noh, H., and M. Hickman. Logit-Based Transit Assignment Considering Capacity Constraints using a Diagonalized Gradient Projection Method. In 12th International *Conference on Advanced Systems for Public Transport*, 2012.

Noh, H., M. Hickman, and A. Khani. Hyperpaths in a Transit Schedule-Based Network. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2284, 2012a, pp. 29-39.

———. Logit-Based Congested Transit Assignment using Hyperpaths on a Scheduled Transit Network. In *The 4th International Symposium on Dynamic Traffic Assignment*, 2012b.

Nuzzolo, A., F. Russo, and U. Crisalli. A Doubly Dynamic Schedule-Based Assignment Model for Transit Networks. *Transportation Science*, Vol. 35, No. 3, 2001, pp. 268-285.

Poon, M. H., S. C. Wong, and C. O. Tong. A Dynamic Schedule-Based Model for Congested Transit Networks. *Transportation Research Part B: Methodological*, Vol. 38, No. 4, 2004, pp. 343-368.

Potts, R. B., and R. M. Oliver. *Flows in Transportation Networks.* Academic Press, 1972.

Prashker, J. N., and S. Bekhor. Congestion, Stochastic, and Similarity Effects in Stochastic User-Equilibrium Models. *Transportation Research Record* 1733, TRB, National Research Council, Washington D.C., 2000, pp. 80–87.

———. Investigation of Stochastic Network Loading Procedures. *Transportation Research Record* 1645, TRB, National Research Council, Washington D.C., 1998, pp. 94–102.

Prato, G. P. Route Choice Modeling: Past, Present and Future Research Directions. *Journal of Choice Modelling*, Vol. 2, No. 1, 2009, pp. 65-100.

Rochau, N., M. G. H. Bell, K. Nökel, and A. Fonzone. Schedule-Based Hyperpath Approaches to Transit Assignment: The Impact of Imperfect Information (Unpublished). In *5th IMA Conference on Mathematics in Transport*, 2010.

Sheffi, Y. *Urban Transportation Networks: Equilibrium Anlaysis with Mathematical Programming Methods.* Prentice-Hall, INC., Englewood Cliffs, New Jersey 07632, 1985.

Smith, M. J. An Algorithm for Solving Asymmetric Equilibrium Problems with A Continous Cost-Flows Function, *Transportation Research Part B: Methodological*, Vol. 17, No. 5, 1983, pp. 365-371.

Spiess, H., and M. Florian. Optimal Strategies: A New Assignment Model for Transit Networks. *Transportation Research Part B: Methodological*, Vol. 23, No. 2, 1989, pp. 83-102.

Sun, C., R. Jayakrishnan, and W. Tsai. Computational Study of a Path-Based Algorithm and its Variants for Static Traffic Assignment. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 1537, 1996, pp. 106-115.

Tong C.O., and A. J. Richardson. A Computer Model for Finding the Time-Dependent Minimum Path in a Transit System with Fixed Schedules. *Journal of Advanced Transportation*, Vol. 18, 1984, pp. 145-161.

Tong, C. O., and S. C. Wong. A Stochastic Transit Assignment Model using a Dynamic Schedule-Based Network. *Transportation Research Part B: Methodological*, Vol. 33, 1999, pp. 107-121.

Tong , C. O., S. C. Wong, M. H. Poon, and M.C. Tan. A Schedule-based Dynamic Transit Network Model – Recent Advances and Prospective Future Research. *Journal of Advanced Transportation,* Vol. 35, No. 2, 2001, pp. 175-195.

Vovsha, P. Application of Cross-Nested Logit Model to Mode Choice in Tel Aviv, Israel, Metropolitan Area. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 1607, 1997, pp. 6-14.

Wu, J. H., M. Florian, and P. Marcotte. Transit Equilibrium Assignment: A Model and Solution Algorithms. *Transportation Science*, Vol. 28, No. 3, 1994, pp. 193-203.

Xu, X., A. Chen, Z. Zhou, and S. Bekhor. Path-Based Algorithms to Solve C-Logit Stochastic User Equilibrium Assignment Problem. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2279, 2012, pp. 21-30.

Zhou, Z., A. Chen. A Self-adaptive Scaling Technique Embedded in the Projection Traffic Assignment Algorithm. *Journal of Eastern Asia Society for Transportation Studies*, Vol. 5, No. 16, 2003, pp. 47–62.

GTFS Data Exchange. www.gtfs-data-exchange.com. Accessed June, 2012.

Google Transit Data Feed. https://code.google.com/p/googletransitdatafeed/wiki/PublicFeeds. Accessed June, 2012.